Lecture I: Introduction to Monte Carlo Methods, Integration and Probability Distributions

Morten Hjorth-Jensen

[1]Department of Physics and Center of Mathematics for Applications
University of Oslo, N-0316 Oslo, Norway

[2]Department of Physics and Astronomy, Michigan State University
East Lansing, Michigan, USA

January 28 - February 2

Outline

1. Introduction to Monte Carlo Methods
2. Probability Distribution Functions
3. Monte Carlo Integration

Outline

1. Introduction to Monte Carlo Methods
2. Probability Distribution Functions
3. Monte Carlo Integration

Outline

1. Introduction to Monte Carlo Methods
2. Probability Distribution Functions
3. Monte Carlo Integration

'Iacta Alea est', the die is cast!

Plan for the lectures

1. January 28: Introduction to Monte Carlo methods, probability distributions and Monte Carlo Integration.
2. January 29: Random numbers, Markov chains, diffusion and the Metropolis algorithm.
3. January 30: Applications in sociology, simulations of phase transitions in physics and quantum physics.
4. All material taken from my text on Computational Physics, see `http://www.uio.no/studier/emner/matnat/fys/FYS3150/h06/undervisningsmateriale/LectureNotes/`.

http://www.iop.org/EJ/journal/CSD

NEW FOR 2006

COMPUTATIONAL SCIENCE&DISCOVERY

Uniquely driven by the aim to publish multidisciplinary scientific advances, together with details of their enabling technologies.

www.iop.org/journals/csd

What is Monte Carlo?

1. **Monte Carlo** methods are nowadays widely used, from the integration of multi-dimensional integrals to solving ab initio problems in chemistry, physics, medicine, biology, or even Dow-Jones forecasting. Computational finance is one of the novel fields where Monte Carlo methods have found a new field of applications, with financial engineering as an emerging field.
2. **Numerical methods** that are known as Monte Carlo methods can be loosely described 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.

Monte Carlo Keywords

Consider it is a numerical experiment

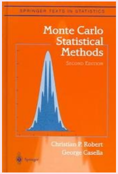
- Be able to generate random variables following a given PDF
- Find a probability distribution function (PDF).
- Sampling rule for accepting a move
- Compute standard deviation and other expectation values
- Techniques for improving errors

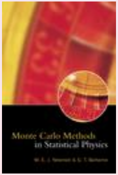
The Plethora of Applications; from the Sciences to Social Studies

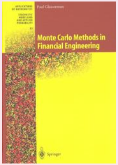
1. Quantum Physics and Chemistry: Variational, Diffusion and Path Integral Monte Carlo
2. Simulations of Phase transitions, classical ones and quantal ones such as superfluidity (quantum liquids)
3. Lattice Quantum-Chromo-Dynamics (QCD), the only way to test the fundamental forces of Nature. (Own dedicated High-Performance-Computing machine).
4. Reconstruction of particle-collisions' paths at for example CERN
5. Solution of Stochastic differential equations
6. Dow-Jones forecasting and financial engineering
7. Modelling electoral patterns
8. Ecological evolution models, percolation, wood fires, earthquakes....and so forth

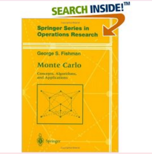
Selected Texts

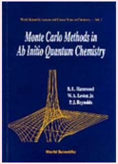
- C. R. Robert and G. Casella,*Monte Carlo Statistical Methods*, Springer, 2nd edition 2004.
- M. E. J. Newman and G. T. Barkema, *Monte Carlo Methods in Statistical Physics*, Oxford University Press, 1999.
- P. Glasserman, *Monte Carlo Methods in Financial Engineering*, Springer, 2003.
- B. L. Hammond, W. A. Lester, Jr., P. J. Reynolds, *Monte Carlo Methods in Ab Initio Quantum Chemistry*, World Scientific, 1994.
- G. S. Fishman, *Monte Carlo Methods, Concepts, Algorithms and Applications*,




Monte Carlo Methods in Statistical Physics
M. E. J. Newman & G. T. Barkema




SEARCH INSIDE!™
Springer Series in Operations Research
George S. Fishman
Monte Carlo
Concepts, Algorithms, and Applications
Springer



Important Application: Monte Carlo Integration

Consider

$$I = \int_0^1 f(x)dx \approx \sum_{i=1}^{N} \omega_i f(x_i),$$

where ω_i are the weights determined by the specific integration method (like Simpson's or Taylor's methods) with x_i the given mesh points. To give you a feeling of how we are to evaluate the above integral using Monte-Carlo, we employ here the crudest possible approach. Later on we will present more refined approaches. This crude approach consists in setting all weights equal 1, $\omega_i = 1$. Recall also that $dx = h = (b - a)/N$ where $b = 1$, $a = 0$ in our case and h is the step size. We can then rewrite the above integral as

$$I = \int_0^1 f(x)dx \approx \frac{1}{N} \sum_{i=1}^{N} f(x_i).$$

Introduce the concept of the average of the function f for a given Probability Distribution Function $p(x)$ as

$$E[f] = \langle f \rangle = \frac{1}{N} \sum_{i=1}^{N} f(x_i)p(x_i),$$

and identify $p(x)$ with the uniform distribution, viz $p(x) = 1$ when $x \in [0, 1]$ and zero for all other values of x.

Monte Carlo Integration

The integral is then the average of f over the interval $x \in [0,1]$

$$I = \int_0^1 f(x)dx \approx E[f] = \langle f \rangle.$$

In addition to the average value $E[f]$ the other important quantity in a Monte-Carlo calculation is the variance σ^2 and the standard deviation σ. We define first the variance of the integral with f for a uniform distribution in the interval $x \in [0,1]$ to be

$$\sigma_f^2 = \frac{1}{N}\sum_{i=1}^{N}(f(x_i) - \langle f \rangle)^2 p(x_i),$$

and inserting the uniform distribution this yields

$$\sigma_f^2 = \frac{1}{N}\sum_{i=1}^{N} f(x_i)^2 - \left(\frac{1}{N}\sum_{i=1}^{N} f(x_i)\right)^2,$$

or

$$\sigma_f^2 = \left(E[f^2] - (E[f])^2\right).$$

which is nothing but a measure of the extent to which f deviates from its average over the region of integration.

But what do we gain by Monte Carlo Integration?

- The trapezoidal rule carries a truncation error $O(h^2)$, with h the step length.
- In general, quadrature rules such as Newton-Cotes have a truncation error which goes like $\sim O(h^k)$, with $k \geq 1$. Recalling that the step size is defined as $h = (b - a)/N$, we have an error which goes like $\sim N^{-k}$.
- Monte Carlo integration is more efficient in higher dimensions. Assume that our integration volume is a hypercube with side L and dimension d. This cube contains hence $N = (L/h)^d$ points and therefore the error in the result scales as $N^{-k/d}$ for the traditional methods.
- The error in the Monte carlo integration is however independent of d and scales as $\sigma \sim 1/\sqrt{N}$, always!
- Comparing this with traditional methods, shows that Monte Carlo integration is more efficient than an order-k algorithm when $d > 2k$

Why Monte Carlo Integration?

An example from quantum mechanics: most problems of interest in e.g., atomic, molecular, nuclear and solid state physics consist of a large number of interacting electrons and ions or nucleons. The total number of particles N is usually sufficiently large that an exact solution cannot be found. Typically, the expectation value for a chosen hamiltonian for a system of N particles is

$$\langle H\rangle =$$

$$\frac{\int d\mathbf{R}_1 d\mathbf{R}_2\dots d\mathbf{R}_N\Psi^*(\mathbf{R_1},\mathbf{R}_2,\dots,\mathbf{R}_N)H(\mathbf{R_1},\mathbf{R}_2,\dots,\mathbf{R}_N)\Psi(\mathbf{R_1},\mathbf{R}_2,\dots,\mathbf{R}_N)}{\int d\mathbf{R}_1 d\mathbf{R}_2\dots d\mathbf{R}_N\Psi^*(\mathbf{R_1},\mathbf{R}_2,\dots,\mathbf{R}_N)\Psi(\mathbf{R_1},\mathbf{R}_2,\dots,\mathbf{R}_N)},$$

an in general intractable problem.

This integral is actually the starting point in a Variational Monte Carlo calculation.

Gaussian quadrature: Forget it! given 10 particles and 10 mesh points for each degree of freedom and an ideal 1 Tflops machine (all operations take the same time), how long will it ta ke to compute the above integral? Lifetime of the universe $T \approx 4.7 \times 10^{17}$s.

The Dimensionality Curse

As an example from the nuclear many-body problem, we have Schrödinger's equation as a differential equation

$$\hat{H}\Psi(\mathbf{r}_1, .., \mathbf{r}_A, \alpha_1, .., \alpha_A) = E\Psi(\mathbf{r}_1, .., \mathbf{r}_A, \alpha_1, .., \alpha_A)$$

where

$$\mathbf{r}_1, .., \mathbf{r}_A,$$

are the coordinates and

$$\alpha_1, .., \alpha_A,$$

are sets of relevant quantum numbers such as spin and isospin for a system of A nucleons ($A = N + Z$, N being the number of neutrons and Z the number of protons).

More on Dimensionality

There are

$$2^A \times \left(\begin{array}{c} A \\ Z \end{array} \right)$$

coupled second-order differential equations in $3A$ dimensions.

For a nucleus like ^{10}Be this number is **215040**. This is a truely challenging many-body problem.

Another classic: Radioactive Decay

Assume that a the time $t = 0$ we have $N(0)$ nuclei of type X which can decay radioactively. At a time $t > 0$ we are left with $N(t)$ nuclei. With a transition probability ω, which expresses the probability that the system will make a transition to another state during a time step of one second, we have the following first-order differential equation

$$dN(t) = -\omega N(t)dt,$$

whose solution is

$$N(t) = N(0)e^{-\omega t},$$

where we have defined the mean lifetime τ of X as

$$\tau = \frac{1}{\omega}.$$

Radioactive Decay

Probability for a decay of a particle during a time step Δt is

$$\frac{\Delta N(t)}{N(t)\Delta t} = -\lambda$$

λ is inversely proportional to the lifetime

- Choose the number of particles $N(t = 0) = N_0$.
- Make a loop over the number of time steps, with maximum time bigger than the number of particles N_0
- At every time step there is a probability λ for decay. Compare this probability with a random number x.
- If $x \leq \lambda$, reduce the number of particles with one i.e., $N = N - 1$. If not, keep the same number of particles till the next time step.
- Increase by one the time step (the external loop)

Radioactive Decay

```
  idum=-1;  // initialise random number generator
  // loop over monte carlo cycles
  // One monte carlo loop is one sample
  for (cycles = 1; cycles <= number_cycles; cycles++){
    n_unstable = initial_n_particles;
    //  accumulate the number of particles per time step per trial
    ncumulative[0] += initial_n_particles;
    // loop over each time step
    for (time=1; time <= max_time; time++){
      // for each time step, we check each particle
      particle_limit = n_unstable;
      for ( np = 1; np <=  particle_limit; np++) {
        if( ran0(&idum) <= decay_probability) {
          n_unstable=n_unstable-1;
        }
      }  // end of loop over particles
      ncumulative[time] += n_unstable;
    }  // end of loop over time steps
  }    // end of loop over MC trials
}   // end mc_sampling function
```

The MC Philosophy in a Nutshell

- Choose the number of Monte Carlo samples N. Think of every sample as an experiment. Make a loop over N. These samples are often called Monte Carlo cycles or just samples.
- Within one experiment you may study a given physical system, say the alpha decay of 100 nuclei every day.
- You need a sampling rule. For this decay you choose a random variable from the uniform distribution with x_i in the interval $x_i \in [0,1]$ by calling a random number generator. This number is compared with your decay probability. If smaller diminish the number of particles, if bigger keep the number. **This is the sampling rule**
- Every experiment has its mean and variance. Find the contribution to the variance and the mean value for every loop contribution.
- After N samplings, compute the final mean value, variance, standard deviation and possibly the covariance.

Probability Distribution Functions PDF

	Discrete PDF	continuous PDF
Domain	$\{x_1, x_2, x_3, \dots, x_N\}$	$[a, b]$
probability	$p(x_i)$	$p(x)dx$
Cumulative	$P_i = \sum_{l=1}^{i} p(x_l)$	$P(x) = \int_a^x p(t)dt$
Positivity	$0 \leq p(x_i) \leq 1$	$p(x) \geq 0$
Positivity	$0 \leq P_i \leq 1$	$0 \leq P(x) \leq 1$
Monotonuous	$P_i \geq P_j$ if $x_i \geq x_j$	$P(x_i) \geq P(x_j)$ if $x_i \geq x_j$
Normalization	$P_N = 1$	$P(b) = 1$

As an example, consider the tossing of two dice, which yields the following possible values

$$[2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12].$$

These values are called the *domain*. To this domain we have the corresponding *probabilities*

$$[1/36, 2/36/3/36, 4/36, 5/36, 6/36, 5/36, 4/36, 3/36, 2/36, 1/36].$$

Expectation Values

- Discrete PDF

$$E[x^k] = \langle x^k \rangle = \frac{1}{N}\sum_{i=1}^{N} x_i^k p(x_i),$$

provided that the sums (or integrals) $\sum_{i=1}^{N} p(x_i)$ converge absolutely (viz , $\sum_{i=1}^{N} |p(x_i)|$ converges)

- Continuous PDF

$$E[x^k] = \langle x^k \rangle = \int_a^b x^k p(x)dx,$$

- Function $f(x)$

$$E[f^k] = \langle f^k \rangle = \int_a^b f^k p(x)dx,$$

- Variance

$$\sigma_f^2 = E[f^2] - (E[f])^2 = \langle f^2 \rangle - \langle f \rangle^2$$

Uniform Distribution

The uniform PDF

$$p(x) = \frac{1}{b-a}\Theta(x-a)\Theta(b-x).$$

It gives for $a = 0, b = 1$ $p(x) = 1$ for $x \in [0,1]$ and zero else. It forms the basis for all generations of random numbers.

Exponential Distribution

The exponential PDF

$$p(x) = \alpha e^{-\alpha x},$$

yielding probabilities different from zero in the interval $[0, \infty)$ and with mean value

$$\mu = \int_0^\infty xp(x)dx = \int_0^\infty x\alpha e^{-\alpha x}dx = \frac{1}{\alpha}$$

and variance

$$\sigma^2 = \int_0^\infty x^2 p(x)dx - \mu^2 = \frac{1}{\alpha^2}.$$

Normal Distribution

The Normal PDF

$$p(x) = \frac{1}{b\sqrt{2\pi}} \exp\left(-\frac{(x-a)^2}{2b^2}\right)$$

with probabilities different from zero in the interval $(-\infty, \infty)$. The integral $\int_{-\infty}^{\infty} \exp\left(-(x^2\right) dx$ appears in many calculations, its value is $\sqrt{\pi}$, a result we will need when we compute the mean value and the variance. The mean value is

$$\mu = \int_0^{\infty} xp(x)dx = \frac{1}{b\sqrt{2\pi}} \int_{-\infty}^{\infty} x \exp\left(-\frac{(x-a)^2}{2b^2}\right) dx,$$

which becomes with a suitable change of variables

$$\mu = \frac{1}{b\sqrt{2\pi}} \int_{-\infty}^{\infty} b\sqrt{2}(a + b\sqrt{2}y) \exp -y^2 dy = a.$$

Normal Distribution, further Properties

Similarly, the variance becomes

$$\sigma^2 = \frac{1}{b\sqrt{2\pi}} \int_{-\infty}^{\infty} (x-\mu)^2 \exp\left(-\frac{(x-a)^2}{2b^2}\right) dx,$$

and inserting the mean value and performing a variable change we obtain

$$\sigma^2 = \frac{1}{b\sqrt{2\pi}} \int_{-\infty}^{\infty} b\sqrt{2}(b\sqrt{2}y)^2 \exp\left(-y^2\right) dy = \frac{2b^2}{\sqrt{\pi}} \int_{-\infty}^{\infty} y^2 \exp\left(-y^2\right) dy,$$

and performing a final integration by parts we obtain the well-known result $\sigma^2 = b^2$.

Normal Distribution, further Properties

It is useful to introduce the standard normal distribution as well, defined by $\mu = a = 0$, viz. a distribution centered around zero and with a variance $\sigma^2 = 1$, leading to

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

The exponential and uniform distributions have simple cumulative functions, whereas the normal distribution does not, being proportional to the so-called error function *erf*(x), given by

$$P(x) = .\frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp\left(-\frac{t^2}{2}\right) dt,$$

which is difficult to evaluate in a quick way. Later we will present an algorithm by Box and Mueller which allows us to compute the cumulative distribution using random variables sampled from the uniform distribution.

Binomial Distribution

The binomial distribution

$$p(x) = \left(\begin{array}{c} n \\ x \end{array} \right) y^x (1-y)^{n-x} \quad x = 0, 1, \dots, n,$$

where y is the probability for a specific event, such as the tossing of a coin or moving left or right in case of a random walker. Note that x is a discrete stochastic variable. The sequence of binomial trials is characterized by the following definitions

- Every experiment is thought to consist of N independent trials.
- In every independent trial one registers if a specific situation happens or not, such as the jump to the left or right of a random walker.
- The probability for every outcome in a single trial has the same value, for example the outcome of tossing a coin is always $1/2$.

In Lecture 3 we will show that the probability distribution for a random walker approaches the binomial distribution.

Properties of the Binomial Distribution

In order to compute the mean and variance we need to recall Newton's binomial formula

$$(a+b)^m = \sum_{n=0}^{m} \left(\begin{array}{c} m \\ n \end{array} \right) a^n b^{m-n},$$

which can be used to show that

$$\sum_{x=0}^{n} \left(\begin{array}{c} n \\ x \end{array} \right) y^x (1-y)^{n-x} = (y+1-y)^n = 1,$$

the PDF is normalized to one.

Properties of the Binomial Distribution

The mean value is

$$\mu = \sum_{x=0}^{n} x \left(\begin{array}{c} n \\ x \end{array} \right) y^x (1-y)^{n-x} = \sum_{x=0}^{n} x \frac{n!}{x!(n-x)!} y^x (1-y)^{n-x},$$

resulting in

$$\mu = \sum_{x=0}^{n} x \frac{(n-1)!}{(x-1)!(n-1-(x-1))!} y^{x-1} (1-y)^{n-1-(x-1)},$$

which we rewrite as

$$\mu = ny \sum_{\nu=0}^{n} \left(\begin{array}{c} n-1 \\ \nu \end{array} \right) y^{\nu} (1-y)^{n-1-\nu} = ny(y+1-y)^{n-1} = ny.$$

The variance is slightly trickier to get. Exercise: show that it reads $\sigma^2 = ny(1-y)$.

Poisson Distribution

Another important distribution with discrete stochastic variables x is the Poisson model, which resembles the exponential distribution and reads

$$p(x) = \frac{\lambda^x}{x!}e^{-\lambda} \; x = 0, 1, \dots, ; \lambda > 0.$$

In this case both the mean value and the variance are easier to calculate,

$$\mu = \sum_{x=0}^{\infty} x \frac{\lambda^x}{x!}e^{-\lambda} = \lambda e^{-\lambda}\sum_{x=1}^{\infty}\frac{\lambda^{x-1}}{(x-1)!} = \lambda,$$

and the variance is $\sigma^2 = \lambda$. Example of applications of the Poisson distribution is the counting of the number of α-particles emitted from a radioactive source in a given time interval. In the limit of $n \rightarrow \infty$ and for small probabilities y, the binomial distribution approaches the Poisson distribution. Setting $\lambda = ny$, with y the probability for an event in the binomial distribution we can show that

$$\lim_{n\rightarrow\infty}\left(\begin{array}{c} n \\ x \end{array}\right) y^x(1-y)^{n-x}e^{-\lambda}\sum_{x=1}^{\infty} = \frac{\lambda^x}{x!}e^{-\lambda},$$

Multivariable Expectation Values

Let us recapitulate some of the above concepts using a discrete PDF (which is what we end up doing anyway on a computer). The mean value of a random variable X with range $x_1, x_2, \ldots, N$ is

$$\langle x \rangle = \mu = \frac{1}{N}\sum_{i=1}^{N} x_i p(x_i),$$

and the variance is

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N} (x_i - \langle x \rangle)^2 p(x_i) = \frac{1}{N}\sum_{i=1}^{N} \langle (x_i - \mu_i)^2 \rangle.$$

Assume now that we have two independent sets of measurements X_1 and X_2 with corresponding mean and variance μ_1 and μ_2 and $\sigma^2_{X_1}$ and $\sigma^2_{X_2}$.

Multivariable Expectation Values

It follows that if we define the new stochastic variable

$$Y = X_1 + X_2,$$

we have

$$\mu_Y = \mu_1 + \mu_2,$$

and

$$\sigma_Y^2 = \langle (Y - \mu_Y)^2 \rangle = \langle (X_1 - \mu_1)^2 \rangle + \langle (X_2 - \mu_2)^2 \rangle + 2\langle (X_1 - \mu_1)(X_2 - \mu_2) \rangle.$$

It is useful to define the so-called covariance, given by

$$cov(X_1, X_2) = \langle (X_1 - \mu_1)(X_2 - \mu_2) \rangle$$

where we consider the averages μ_1 and μ_2 as the outcome of two separate measurements. The covariance measures thus the degree of correlation between variables. We can then rewrite the variance of Y as

$$\sigma_Y^2 = \sum_{j=1}^{2} \langle (X_j - \mu_j)^2 \rangle + 2cov(X_1, X_2),$$

which in our notation becomes

$$\sigma_Y^2 = \sigma_{X_1}^2 + \sigma_{X_2}^2 + 2cov(X_1, X_2).$$

If X_1 and X_2 are two independent variables we can show that the covariance is zero, but one cannot deduce from a zero covariance whether the variables are independent or not. If our random variables which we generate are truely random numbers, then the covariance should be zero.

A way to measure the correlation between two sets of stochastic variables is the so-called correlation function $\rho(X_1, X_2)$ defined as

$$\rho(X_1, X_2) = \frac{cov(X_1, X_2)}{\sqrt{\langle \sigma^2 \rangle_{X_1} \langle \sigma^2 \rangle_{X_2}}}.$$

Obviously, if the covariance is zero due to the fact that the variables are independent, then the correlation is zero. This quantity is often called the correlation coefficient between X_1 and X_2. We can extend this analysis to a set of stochastic variables $Y = (X_1 + X_2 + \cdots + X_N)$. We now assume that we have N different measurements of the mean and variance of a given variable. Each measurement consists again of N measurements, although we could have chosen the latter to be different from N. The total mean value is defined as

$$\langle \mu_Y \rangle = \sum_{i=1}^{N} \langle \mu_i \rangle.$$

The total variance is however now defined as

$$\sigma_Y^2 = \langle (Y - \mu_Y)^2 \rangle = \sum_{j=1}^{N} \langle (X_j - \mu_j) \rangle^2 = \sum_{j=1}^{N} \sigma_{X_j}^2 + 2 \sum_{j<k}^{N} \langle (X_j - \mu_j) \rangle \langle (X_k - \mu_k) \rangle,$$

or

$$\sigma_Y^2 = \sum_{j=1}^{N} \sigma_{X_j}^2 + 2 \sum_{j<k}^{N} cov(X_j, X_k).$$

Covariance

If the variables are independent, the covariance is zero and the variance is reduced to

$$\sigma_Y^2 = \sum_{j=1}^{N} \sigma_{X_j}^2,$$

and if we assume that all sets of measurements produce the same variance σ^2, we end up with

$$\sigma_Y^2 = N\sigma^2.$$

In Lecture 5 we will discuss a very important class of correlation functions (another application of the covariance), the so-called time-correlation functions. This are important quantities in our studies of equilibrium properties,

$$\phi(t) = \int dt' \left[\mathcal{M}(t') - \langle\mathcal{M}\rangle\right]\left[\mathcal{M}(t'+t) - \langle\mathcal{M}\rangle\right].$$

From Onsager regression hypothesis, we have that in the long time limit, the variables $\mathcal{M}(t'+t)$ and $\mathcal{M}(t)$ eventually become uncorrelated from each other so that the time correlation function becomes zero. The system has then reached its most likely state.

Central Limit Theorem

Suppose we have a PDF $p(x)$ from which we generate a series N of averages $\langle x_i \rangle$. Each mean value $\langle x_i \rangle$ is viewed as the average of a specific measurement, e.g., throwing dice 100 times and then taking the average value, or producing a certain amount of random numbers. For notational ease, we set $\langle x_i \rangle = x_i$ in the discussion which follows.

If we compute the mean z of N such mean values x_i

$$z = \frac{x_1 + x_2 + \cdots + x_N}{N},$$

the question we pose is which is the PDF of the new variable z.

Central Limit Theorem

The probability of obtaining an average value z is the product of the probabilities of obtaining arbitrary individual mean values x_i, but with the constraint that the average is z. We can express this through the following expression

$$\tilde{p}(z) = \int dx_1 p(x_1) \int dx_2 p(x_2) \dots \int dx_N p(x_N) \delta(z - \frac{x_1 + x_2 + \dots + x_N}{N}),$$

where the δ-function enbodies the constraint that the mean is z. All measurements that lead to each individual x_i are expected to be independent, which in turn means that we can express $\tilde{p}$ as the product of individual $p(x_i)$.

Central Limit Theorem

If we use the integral expression for the δ-function

$$\delta(z-\frac{x_1+x_2+\cdots+x_N}{N})=\frac{1}{2\pi}\int_{-\infty}^{\infty}dqe^{\left(iq(z-\frac{x_1+x_2+\cdots+x_N}{N})\right)},$$

and inserting $e^{i\mu q-i\mu q}$ where μ is the mean value we arrive at

$$\tilde{p}(z)=\frac{1}{2\pi}\int_{-\infty}^{\infty}dqe^{(iq(z-\mu))}\left[\int_{-\infty}^{\infty}dxp(x)e^{(iq(\mu-x)/N)}\right]^N,$$

with the integral over x resulting in

$$\int_{-\infty}^{\infty}dxp(x)\exp\left(iq(\mu-x)/N\right)=\int_{-\infty}^{\infty}dxp(x)\left[1+\frac{iq(\mu-x)}{N}-\frac{q^2(\mu-x)^2}{2N^2}+\dots\right].$$

Central Limit Theorem

The second term on the rhs disappears since this is just the mean and employing the definition of σ^2 we have

$$\int_{-\infty}^{\infty} dx p(x) e^{(iq(\mu - x)/N)} = 1 - \frac{q^2\sigma^2}{2N^2} + \ldots,$$

resulting in

$$\left[\int_{-\infty}^{\infty} dx p(x) \exp\left(iq(\mu - x)/N\right)\right]^N \approx \left[1 - \frac{q^2\sigma^2}{2N^2} + \ldots\right]^N,$$

and in the limit $N \to \infty$ we obtain

$$\tilde{p}(z) = \frac{1}{\sqrt{2\pi}(\sigma/\sqrt{N})} \exp\left(-\frac{(z-\mu)^2}{2(\sigma/\sqrt{N})^2}\right),$$

which is the normal distribution with variance $\sigma_N^2 = \sigma^2/N$, where σ is the variance of the PDF $p(x)$ and μ is also the mean of the PDF $p(x)$.

Central Limit Theorem

Thus, the central limit theorem states that the PDF $\tilde{p}(z)$ of the average of N random values corresponding to a PDF $p(x)$ is a normal distribution whose mean is the mean value of the PDF $p(x)$ and whose variance is the variance of the PDF $p(x)$ divided by N, the number of values used to compute z.
The theorem is satisfied by a large class of PDFs. Note however that for a finite N, it is not always possible to find a closed expression for $\tilde{p}(x)$. The central limit theorem leads then to the well-known expression for the standard deviation, given by

$$\sigma_N = \frac{\sigma}{\sqrt{N}}.$$

The latter is true only if the average value is known exactly. This is obtained in the limit $N \rightarrow \infty$ only.

Monte Carlo Integration

With the uniform distribution $p(x) = 1$ for $x \in [0, 1]$ and zero else

$$I = \int_0^1 f(x)dx \approx \frac{1}{N}\sum_{i=1}^{N} f(x_i),$$

$$I = \int_0^1 f(x)dx \approx E[f] = \langle f \rangle.$$

$$\sigma_f^2 = \frac{1}{N}\sum_{i=1}^{N} f(x_i)^2 - \left(\frac{1}{N}\sum_{i=1}^{N} f(x_i)\right)^2,$$

or

$$\sigma_f^2 = E[f^2] - (E[f])^2 = \left(\langle f^2 \rangle - \langle f \rangle^2\right).$$

Brute Force Algorithm for Monte Carlo Integration

- Choose the number of Monte Carlo samples N.
- Make a loop over N and for every step generate a random number x_i in the interval $x_i \in [0, 1]$ by calling a random number generator.
- Use this number to compute $f(x_i)$.
- Find the contribution to the variance and the mean value for every loop contribution.
- After N samplings, compute the final mean value and the standard deviation

Brute Force Integration

```
// crude mc function to calculate pi
     int i, n;
     long idum;
     double crude_mc, x, sum_sigma, fx, variance;
     cout << "Read in the number of Monte-Carlo samples" << endl;
     cin >> n;
     crude_mc = sum_sigma=0. ; idum=-1 ;
//    evaluate the integral with the a crude Monte-Carlo method
      for ( i = 1;  i <= n; i++){
           x=ran0(&idum);
           fx=func(x);
           crude_mc += fx;
           sum_sigma += fx*fx;
      }
      crude_mc = crude_mc/((double) n );
      sum_sigma = sum_sigma/((double) n );
      variance=sum_sigma-crude_mc*crude_mc;
```

Code at `http://folk.uio.no/mhjensen/fys3150/2005/programs/chapter8/example1.cpp`.

Or: another Brute Force Integration

```
// crude mc function to calculate pi
int main()
{
  const int n = 1000000;
  double x, fx, pi, invers_period, pi2;
  int i;
  invers_period = 1./RAND_MAX;
  srand(time(NULL));
  pi = pi2 = 0.;
  for (i=0; i<n;i++)
    {
      x = double(rand())*invers_period;
      fx = 4./(1+x*x);
      pi += fx;
      pi2 += fx*fx;
    }
  pi /= n;  pi2 = pi2/n - pi*pi;
  cout << "pi=" << pi << " sigma^2=" << pi2 << endl;
  return 0;
}
```

Brute Force Integration

Note the call to a function which generates random numbers according to the uniform distribution

```
     long idum;
     idum=-1 ;
     .....
     x=ran0(&idum);
     ....
```

or

```
  ...
  invers_period = 1./RAND_MAX;
  srand(time(NULL));
  ...
  x = double(rand())*invers_period;
```

Results of Brute Force Integration

N	I	σ_N
10	3.10263E+00	3.98802E-01
100	3.02933E+00	4.04822E-01
1000	3.13395E+00	4.22881E-01
10000	3.14195E+00	4.11195E-01
100000	3.14003E+00	4.14114E-01
1000000	3.14213E+00	4.13838E-01
10000000	3.14177E+00	4.13523E-01
10^9	3.14162E+00	4.13581E-01

We note that as N increases, the integral itself never reaches more than an agreement to the fourth or fifth digit. The variance also oscillates around its exact value $4.13581E-01$. Note well that the variance need not be zero but one can, with appropriate redefinitions of the integral be made smaller. A smaller variance yields also a smaller standard deviation.

Acceptance-Rejection Method

This is a rather simple and appealing method after von Neumann. Assume that we are looking at an interval $x \in [a,b]$, this being the domain of the PDF $p(x)$. Suppose also that the largest value our distribution function takes in this interval is M, that is

$$p(x) \leq M \qquad x \in [a,b].$$

Then we generate a random number x from the uniform distribution for $x \in [a,b]$ and a corresponding number s for the uniform distribution between $[0,M]$. If

$$p(x) \geq s,$$

we accept the new value of x, else we generate again two new random numbers x and s and perform the test in the latter equation again.

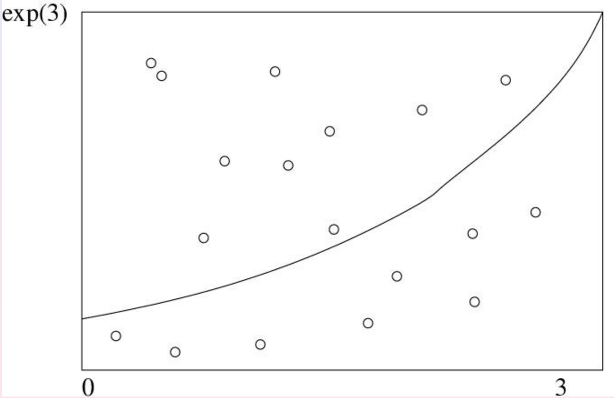
Acceptance-Rejection Method

As an example, consider the evaluation of the integral

$$I = \int_0^3 \exp{(x)} dx.$$

Obviously to derive it analytically is much easier, however the integrand could pose some more difficult challenges. The aim here is simply to show how to implent the acceptance-rejection algorithm. The integral is the area below the curve $f(x) = \exp{(x)}$. If we uniformly fill the rectangle spanned by $x \in [0, 3]$ and $y \in [0, \exp{(3)}]$, the fraction below the curve obatained from a uniform distribution, and multiplied by the area of the rectangle, should approximate the chosen integral. It is rather easy to implement this numerically, as shown in the following code.

Simple Plot of the Accept-Reject Method



Acceptance-Rejection Method

```
//   Loop over Monte Carlo trials n
     integral =0.;
     for ( int i = 1;  i <= n; i++){
//   Finds a random value for x in the interval [0,3]
          x = 3*ran0(&idum);
//   Finds y-value between [0,exp(3)]
          y = exp(3.0)*ran0(&idum);
//   if the value of y at exp(x) is below the curve, we accept
          if ( y  < exp(x)) s = s+ 1.0;
//   The integral is area enclosed below the line f(x)=exp(x)
    }
//  Then we multiply with the area of the rectangle and
//  divide by the number of cycles
    Integral = 3.*exp(3.)*s/n
```

Transformation of Variables

The starting point is always the uniform distribution

$$p(x)dx = \left\{ \begin{array}{cc} dx & 0 \leq x \leq 1 \\ 0 & else \end{array} \right.$$

with $p(x) = 1$ and satisfying

$$\int_{-\infty}^{\infty} p(x)dx = 1.$$

All random number generators provided in the program library generate numbers in this domain.
When we attempt a transformation to a new variable $x \rightarrow y$ we have to conserve the probability

$$p(y)dy = p(x)dx,$$

which for the uniform distribution implies

$$p(y)dy = dx.$$

Transformation of Variables

Let us assume that $p(y)$ is a PDF different from the uniform PDF $p(x) = 1$ with $x \in [0, 1]$. If we integrate the last expression we arrive at

$$x(y) = \int_0^y p(y')dy',$$

which is nothing but the cumulative distribution of $p(y)$, i.e.,

$$x(y) = P(y) = \int_0^y p(y')dy'.$$

This is an important result which has consequences for eventual improvements over the brute force Monte Carlo.

Example 1, a general Uniform Distribution

Suppose we have the general uniform distribution

$$p(y)dy = \begin{cases} \frac{dy}{b-a} & a \leq y \leq b \\ 0 & else \end{cases}$$

If we wish to relate this distribution to the one in the interval $x \in [0, 1]$ we have

$$p(y)dy = \frac{dy}{b-a} = dx,$$

and integrating we obtain the cumulative function

$$x(y) = \int_a^y \frac{dy'}{b-a},$$

yielding

$$y = a + (b-a)x,$$

a well-known result!

Example 2, from Uniform to Exponential

Assume that

$$p(y) = e^{-y},$$

which is the exponential distribution, important for the analysis of e.g., radioactive decay. Again, $p(x)$ is given by the uniform distribution with $x \in [0,1]$, and with the assumption that the probability is conserved we have

$$p(y)dy = e^{-y}dy = dx,$$

which yields after integration

$$x(y) = P(y) = \int_0^y \exp(-y')dy' = 1 - \exp(-y),$$

or

$$y(x) = -ln(1-x).$$

This gives us the new random variable y in the domain $y \in [0, \infty)$ determined through the random variable $x \in [0,1]$ generated by our favorite random generator.

Example 2, from Uniform to Exponential

This means that if we can factor out $\exp{(-y)}$ from an integrand we may have

$$I = \int_0^{\infty} F(y)dy = \int_0^{\infty} \exp{(-y)}G(y)dy$$

which we rewrite as

$$\int_0^{\infty} \exp{(-y)}G(y)dy = \int_0^{\infty} \frac{dx}{dy}G(y)dy \approx \frac{1}{N}\sum_{i=1}^{N} G(y(x_i)),$$

where x_i is a random number in the interval [0,1].

Note that in practical implementations, our random number generators for the uniform distribution never return exactly 0 or 1, but we we may come very close. We should thus in principle set $x \in (0,1)$.

Example 2, from Uniform to Exponential

The algorithm is rather simple. In the function which sets up the integral, we simply need the random number generator for the uniform distribution in order to obtain numbers in the interval [0,1]. We obtain y by the taking the logarithm of $(1 - x)$. Our calling function which sets up the new random variable y may then include statements like

```
.....
idum=-1;
x=ran0(&idum);
y=-log(1.-x);
.....
```

Example 3

Another function which provides an example for a PDF is

$$p(y)dy = \frac{dy}{(a+by)^n},$$

with $n > 1$. It is normalizable, positive definite, analytically integrable and the integral is invertible, allowing thereby the expression of a new variable in terms of the old one. The integral

$$\int_0^\infty \frac{dy}{(a+by)^n} = \frac{1}{(n-1)ba^{n-1}},$$

gives

$$p(y)dy = \frac{(n-1)ba^{n-1}}{(a+by)^n} dy,$$

which in turn gives the cumulative function

$$x(y) = P(y) = \int_0^y \frac{(n-1)ba^{n-1}}{(a+bx)^n} dy' =,$$

resulting in

$$y = \frac{a}{b}\left((1-x)^{-1/(n-1)} - 1\right).$$

Example 4, from Uniform to Normal

For the normal distribution, expressed here as

$$g(x,y) = \exp\left(-(x^2+y^2)/2\right)dxdy.$$

it is rather difficult to find an inverse since the cumulative distribution is given by the error function $erf(x)$.
If we however switch to polar coordinates, we have for x and y

$$r = \left(x^2+y^2\right)^{1/2} \qquad \theta = tan^{-1}\frac{x}{y},$$

resulting in

$$g(r,\theta) = r\exp\left(-r^2/2\right)drd\theta,$$

where the angle θ could be given by a uniform distribution in the region $[0,2\pi]$.

Following example 1 above, this implies simply multiplying random numbers $x \in [0,1]$ by 2π.

Example 4, from Uniform to Normal

The variable r, defined for $r \in [0, \infty)$ needs to be related to to random numbers $x' \in [0, 1]$. To achieve that, we introduce a new variable

$$u = \frac{1}{2}r^2,$$

and define a PDF

$$\exp(-u)du,$$

with $u \in [0, \infty)$. Using the results from example 2, we have that

$$u = -ln(1 - x'),$$

where x' is a random number generated for $x' \in [0, 1]$. With

$$x = rcos(\theta) = \sqrt{2u}cos(\theta),$$

and

$$y = rsin(\theta) = \sqrt{2u}sin(\theta),$$

we can obtain new random numbers x, y through

$$x = \sqrt{-2ln(1 - x')}cos(\theta),$$

and

$$y = \sqrt{-2ln(1 - x')}sin(\theta),$$

with $x' \in [0, 1]$ and $\theta \in 2\pi[0, 1]$.

Example 4, from Uniform to Normal

A function which yields such random numbers for the normal distribution would include statements like

```
.....
idum=-1;
radius=sqrt(-2*ln(1.-ran0(idum)));
theta=2*pi*ran0(idum);
x=radius*cos(theta);
y=radius*sin(theta);
.....
```

Box-Mueller Method for Normal Deviates

```
// random numbers with gaussian distribution
double gaussian_deviate(long * idum)
{
  static int iset = 0;
  static double gset;
  double fac, rsq, v1, v2;
  if ( idum < 0) iset =0;
  if (iset == 0) {
    do {
      v1 = 2.*ran0(idum) -1.0;
      v2 = 2.*ran0(idum) -1.0;
      rsq = v1*v1+v2*v2;
    } while (rsq >= 1.0 || rsq == 0.);
    fac = sqrt(-2.*log(rsq)/rsq);
    gset = v1*fac;
    iset = 1;
    return v2*fac;
  } else {
    iset =0;
    return gset;
  }
```

Importance Sampling

With the aid of the above variable transformations we address now one of the most widely used approaches to Monte Carlo integration, namely importance sampling. Let us assume that $p(y)$ is a PDF whose behavior resembles that of a function F defined in a certain interval $[a, b]$. The normalization condition is

$$\int_a^b p(y)dy = 1.$$

We can rewrite our integral as

$$I = \int_a^b F(y)dy = \int_a^b p(y)\frac{F(y)}{p(y)}dy.$$

Importance Sampling

Since random numbers are generated for the uniform distribution $p(x)$ with $x \in [0,1]$, we need to perform a change of variables $x \rightarrow y$ through

$$x(y) = \int_a^y p(y')dy',$$

where we used

$$p(x)dx = dx = p(y)dy.$$

If we can invert $x(y)$, we find $y(x)$ as well.

Importance Sampling

With this change of variables we can express the integral of Eq. (61) as

$$I = \int_a^b p(y)\frac{F(y)}{p(y)}dy = \int_a^b \frac{F(y(x))}{p(y(x))}dx,$$

meaning that a Monte Carlo evalutaion of the above integral gives

$$\int_a^b \frac{F(y(x))}{p(y(x))}dx = \frac{1}{N}\sum_{i=1}^{N}\frac{F(y(x_i))}{p(y(x_i))}.$$

The advantage of such a change of variables in case $p(y)$ follows closely F is that the integrand becomes smooth and we can sample over relevant values for the integrand. It is however not trivial to find such a function p. The conditions on p which allow us to perform these transformations are

1. p is normalizable and positive definite,
2. it is analytically integrable and
3. the integral is invertible, allowing us thereby to express a new variable in terms of the old one.

Importance Sampling

The algorithm for this procedure is

- Use the uniform distribution to find the random variable y in the interval [0,1]. $p(x)$ is a user provided PDF.
- Evaluate thereafter

$$I = \int_a^b F(x)dx = \int_a^b p(x)\frac{F(x)}{p(x)}dx,$$

by rewriting

$$\int_a^b p(x)\frac{F(x)}{p(x)}dx = \int_a^b \frac{F(x(y))}{p(x(y))}dy,$$

since

$$\frac{dy}{dx} = p(x).$$

- Perform then a Monte Carlo sampling for

$$\int_a^b \frac{F(x(y))}{p(x(y))}dy, \approx \frac{1}{N}\sum_{i=1}^{N}\frac{F(x(y_i))}{p(x(y_i))},$$

with $y_i \in [0,1]$,

- Evaluate the variance

Demonstration of Importance Sampling

$$I = \int_0^1 F(x)dx = \int_0^1 \frac{1}{1+x^2}dx = \frac{\pi}{4}.$$

We choose the following PDF (which follows closely the function to integrate)

$$p(x) = \frac{1}{3}(4-2x) \qquad \int_0^1 p(x)dx = 1,$$

resulting

$$\frac{F(0)}{p(0)} = \frac{F(1)}{p(1)} = \frac{3}{4}.$$

Check that it fullfils the requirements of a PDF. We perform then the change of variables (via the Cumulative function)

$$y(x) = \int_0^x p(x')dx' = \frac{1}{3}x\,(4-x)\,,$$

or

$$x = 2 - (4-3y)^{1/2}$$

We have that when $y = 0$ then $x = 0$ and when $y = 1$ we have $x = 1$.

Simple Code

```
//   evaluate the integral with importance sampling
     for ( int i = 1;  i <= n; i++){
       x = ran0(&idum);  // random numbers in [0,1]
       y = 2 - sqrt(4-3*x);  // new random numbers
       fy=3*func(y)/(4-2*y); // weighted function
       int_mc += fy;
       sum_sigma += fy*fy;
     }
     int_mc = int_mc/((double) n );
     sum_sigma = sum_sigma/((double) n );
     variance=(sum_sigma-int_mc*int_mc);
```

Code at `http://folk.uio.no/mhjensen/fys3150/2005/programs/chapter8/example2.cpp`.

Test Runs and Comparison with Brute Force for $\pi = 3.14159$

The suffix *cr* stands for the brute force approach while *is* stands for the use of importance sampling. All calculations use ran0 as function to generate the uniform distribution.

N	I_{cr}	σ_{cr}	I_{is}	σ_{is}
10000	3.13395E+00	4.22881E-01	3.14163E+00	6.49921E-03
100000	3.14195E+00	4.11195E-01	3.14163E+00	6.36837E-03
1000000	3.14003E+00	4.14114E-01	3.14128E+00	6.39217E-03
10000000	3.14213E+00	4.13838E-01	3.14160E+00	6.40784E-03

Multidimensional Integrals

When we deal with multidimensional integrals of the form

$$I = \int_0^1 dx_1 \int_0^1 dx_2 \dots \int_0^1 dx_d g(x_1, \dots, x_d),$$

with x_i defined in the interval $[a_i, b_i]$ we would typically need a transformation of variables of the form

$$x_i = a_i + (b_i - a_i)t_i,$$

if we were to use the uniform distribution on the interval $[0, 1]$. In this case, we need a Jacobi determinant

$$\prod_{i=1}^{d}(b_i - a_i),$$

and to convert the function $g(x_1, \dots, x_d)$ to

$$g(x_1, \dots, x_d) \rightarrow g(a_1 + (b_1 - a_1)t_1, \dots, a_d + (b_d - a_d)t_d).$$

Example: 6-dimensional Integral

As an example, consider the following six-dimensional integral

$$\int_{-\infty}^{\infty} \mathbf{dxdy} g(\mathbf{x}, \mathbf{y}),$$

where

$$g(\mathbf{x}, \mathbf{y}) = \exp\left(-\mathbf{x}^2 - \mathbf{y}^2 - (\mathbf{x} - \mathbf{y})^2/2\right),$$

with $d = 6$.

Example: 6-dimensional Integral

We can solve this integral by employing our brute force scheme, or using importance sampling and random variables distributed according to a gaussian PDF. For the latter, if we set the mean value $\mu = 0$ and the standard deviation $\sigma = 1/\sqrt{2}$, we have

$$\frac{1}{\sqrt{\pi}} \exp{(-x^2)},$$

and through

$$\pi^3 \int \prod_{i=1}^{6} \left(\frac{1}{\sqrt{\pi}} \exp{(-x_i^2)} \right) \exp{(-(\mathbf{x}-\mathbf{y})^2/2)} dx_1 .... dx_6,$$

we can rewrite our integral as

$$\int f(x_1, \dots, x_d) F(x_1, \dots, x_d) \prod_{i=1}^{6} dx_i,$$

where f is the gaussian distribution.

Brute Force I

```
.....
//   evaluate the integral without importance sampling
//   Loop over Monte Carlo Cycles
     for ( int i = 1;  i <= n; i++){
//   x[] contains the random numbers for all dimensions
       for (int j = 0; j< 6; j++) {
           x[j]=-length+2*length*ran0(&idum);
       }
       fx=brute_force_MC(x);
       int_mc += fx;
       sum_sigma += fx*fx;
     }
     int_mc = int_mc/((double) n );
     sum_sigma = sum_sigma/((double) n );
     variance=sum_sigma-int_mc*int_mc;
......
```

Brute Force II

```
double  brute_force_MC(double *x)
{
   double a = 1.; double b = 0.5;
// evaluate the different terms of the exponential
   double xx=x[0]*x[0]+x[1]*x[1]+x[2]*x[2];
   double yy=x[3]*x[3]+x[4]*x[4]+x[5]*x[5];
   double xy=pow((x[0]-x[3]),2)+pow((x[1]-x[4]),2)+pow((x[2]-x[5]),2);
   return exp(-a*xx-a*yy-b*xy);
```

Full code at `http://folk.uio.no/mhjensen/fys3150/2005/programs/chapter8/example3.cpp`.

Importance Sampling I

```
..........
//   evaluate the integral with importance sampling
     for ( int i = 1;  i <= n; i++){
//   x[] contains the random numbers for all dimensions
       for (int j = 0; j < 6; j++) {
 x[j] = gaussian_deviate(&idum)*sqrt2;
       }
       fx=gaussian_MC(x);
       int_mc += fx;
       sum_sigma += fx*fx;
     }
     int_mc = int_mc/((double) n );
     sum_sigma = sum_sigma/((double) n );
     variance=sum_sigma-int_mc*int_mc;
.............
```

Importance Sampling II

```
// this function defines the integrand to integrate

double  gaussian_MC(double *x)
{
   double a = 0.5;
// evaluate the different terms of the exponential
   double xy=pow((x[0]-x[3]),2)+pow((x[1]-x[4]),2)+pow((x[2]-x[5]),2);
   return exp(-a*xy);
} // end function for the integrand
```

Full code at `http://folk.uio.no/mhjensen/fys3150/2005/programs/chapter8/example4.cpp`.

Test Runs for six-dimensional Integral

Results for as function of number of Monte Carlo samples N. The exact answer is $I \approx 10.9626$ for the integral. The suffix *cr* stands for the brute force approach while *gd* stands for the use of a Gaussian distribution function. All calculations use ran0 as function to generate the uniform distribution.

N	I_{cr}	I_{gd}
10000	1.15247E+01	1.09128E+01
100000	1.29650E+01	1.09522E+01
1000000	1.18226E+01	1.09673E+01
10000000	1.04925E+01	1.09612E+01

Going Parallel with MPI

Task parallelism the work of a global problem can be divided into a number of independent tasks, which rarely need to synchronize. Monte Carlo simulation is one example. It is almost embarrassingly trivial to parallelize Monte Carlo codes. MPI is a message-passing library where all the routines have corresponding C/C++-binding

```
MPI_Command_name
```

and Fortran-binding (routine names are in uppercase, but can also be in lower case)

```
MPI_COMMAND_NAME
```

Computing the 6-dimensional Integral in Parallel

```
#include "mpi.h"
#include <stdio.h>
int main (int nargs, char* args)
{
Declarations ....
 MPI_Init (&nargs, &args);
 MPI_Comm_size (MPI_COMM_WORLD, &size);
 MPI_Comm_rank (MPI_COMM_WORLD, &iam);
 ....
 no_intervalls = mcs/size;
 myloop_begin = iam*no_intervalls + 1;
 myloop_end = (iam+1)*no_intervalls;
```

Computing the 6-dimensional Integral in Parallel

```
for ( int i = myloop_begin;  i <= myloop_end; i++){
//  x[] contains the random numbers for all dimensi
   for (int j = 0; j < 6; j++) {
      x[j] = gaussian_deviate(&idum)*sqrt2;
   }
   fx=gaussian_MC(x);
   average[0] += fx;
   average[1] += fx*fx;
 }
MPI_reduce(average, total_average, 2, MPI_DOUBLE,
            MPI_SUM, 0, MPI_COMM_WORLD)
//print results
MPI_Finalize ();
```

Full code at `http://folk.uio.no/mhjensen/fys3150/2005/programs/chapter8/example6.cpp`.

Exercise

(a) Calculate the integral

$$I = \int_0^1 e^{-x^2} dx,$$

using brute force Monte Carlo with $p(x) = 1$ and importance sampling with $p(x) = ae^{-x}$ where a is a constant.

(b) Calculate the integral

$$I = \int_0^\pi \frac{1}{x^2 + cos^2(x)} dx,$$

with $p(x) = ae^{-x}$ where a is a constant. Determine the value of a which minimizes the variance.

(c) Try to parallelize the code as well.