Monte Carlo Basics

§1 Introduction

WHAT IS THE MONTE CARLO METHOD?

- **Monte Carlo (MC) method**: A computational method that utilizes random numbers.
- Two major applications of the MC method:
  1. Multidimensional integrations (e.g., statistical mechanics in physics);
  2. Simulation of stochastic natural phenomena (e.g., stock price).
- **Numerical vs. MC Integration**

  The simplest numerical integration of a one-dimensional function, $f(x)$, in the range, $[a, b]$, samples function values at regular intervals, $f(x_n = a + (n - 1/2)\Delta x)$ ($n = 1, ..., M$; $\Delta x = (b - a)/M$), see the figure below (this is just an average of $(b - a)f(x_n)$):

$$\int_a^b dx f(x) \cong \sum_{n=1}^{M} \Delta x f(x_n).$$ [1]

The MC method instead generates a sequence of random numbers, r_n, in the range, $[a, b]$, and approximates the integral as (the only difference being random instead of regular points)

$$\int_a^b dx f(x) \cong \sum_{n=1}^{M} \frac{b-a}{M} f(r_n).$$

Both methods may look similar in this one-dimensional example.

Consider now an integration of an N-dimensional function, $f(x_1, ..., x_N)$, in a unit hypercube, $[0, 1]^N$,

$$\int_0^1 dx_1 \cdots \int_0^1 dx_N f(x_1, \ldots, x_N).$$

For example, even a crude numerical integration with 10 function evaluations for each of the N coordinates requires 10^N function evaluations. This is intractable for a large N, say $N = 10^6$. The error of the MC estimates scales with the number, M, of the sample points as $1/M^{1/2}$ (to be proven later). The MC method provides an approximate estimation of the integral with increased accuracy with an increased number of sample points.

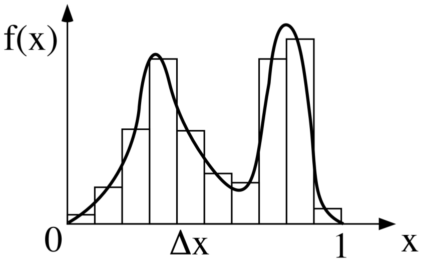


Figure: Numerical integration of a one-dimensional function, $f(x)$.

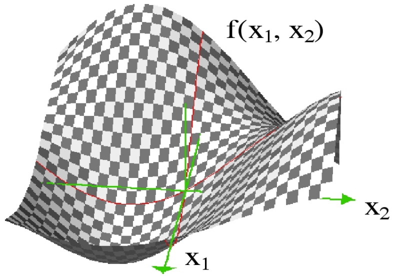
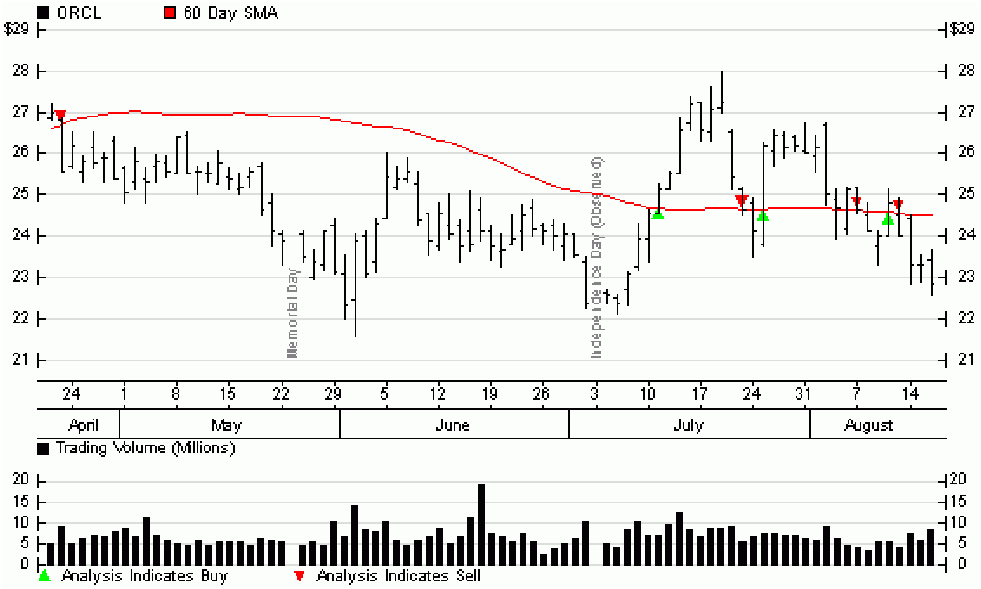


Figure: Numerical integration of a two-dimensional function, $f(x)$.

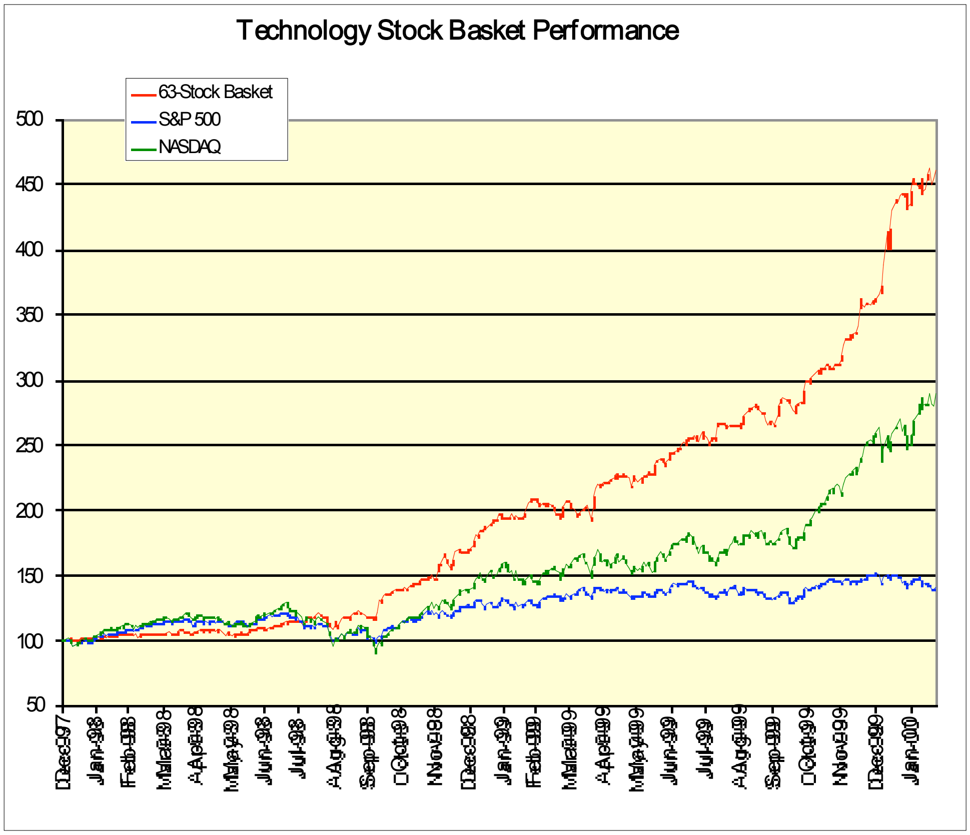
[1] This is the simplest numerical integration with an error on the order of $O(\Delta x)$. For the integration of smooth functions, higher-order numerical integration methods with smaller errors can be used, such as the $O(\Delta x^2)$ trapezoid rule and the $O(\Delta x^4)$ Simpson's rule.

- **MC simulation of stochastic natural phenomena**

  (Example) stock price
  The MC method simulates a stochastic natural phenomenon by generating random numbers.

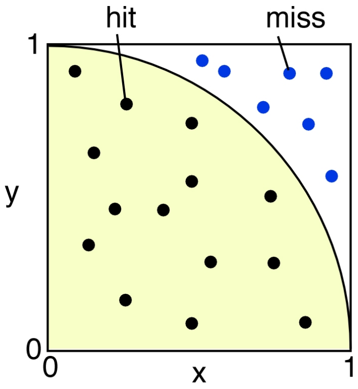


Stock portfolio trading at Quantlab Financial LLC (courtesy of Dr. Andrey Omeltchenko, 1997 graduate of the Collaboratory for Advanced Computing and Simulations)



MC INTEGRATION 1: HIT AND MISS

Consider a square, $\{(x, y) \mid 0 \leq x < 1, 0 \leq y < 1\}$, of unit area. Consider a subspace inside this square, $\{(x, y) \mid 0 \leq x < 1, 0 \leq y < 1, x^2 + y^2 \leq 1\}$. This subspace is a quarter of a unit circle, so that its area is $1/4 \times \pi \times 1^2 = \pi/4$.



The subspace area is estimated by the MC method as follows. Generate N_{try} pairs of random numbers, (x, y), in the range [0, 1]. For each pair, test the condition, $x^2 + y^2 \leq 1$. If this is true, then the pair hits the subspace and we increment the number, N_{hit}, of hits. The area is estimated by

$$N_{\mathrm{hit}}/N_{\mathrm{try}} \times \text{the area of the square} = N_{\mathrm{hit}}/N_{\mathrm{try}}$$

The value of π is estimated by multiplying this value by four.

- Program: `hit.c`

```
/* Monte Carlo integration of PI by hit and miss */
#include <stdio.h>
#include <stdlib.h>
main() {
  double x, y, pi;
  int hit = 0, try, ntry;
  printf("Input the number of MC trials\n");
  scanf("%d",&ntry);
  srand((unsigned)time((long *)0));
  for (try=0; try<ntry; try++) {
    x = rand()/(float)RAND_MAX;
    y = rand()/(float)RAND_MAX;
    if (x*x+y*y < 1.0) ++hit;
  }
  pi = 4.0*hit/ntry;
  printf("MC estimate for PI = %f\n", pi);
}
```

```
> cc hit.c
```

```
> a.out
Input the number of MC trials
10000
Ntry Hit = 10000 7874
MC estimate for PI = 3.149600
```

```
> a.out
Input the number of MC trials
10000
Ntry Hit = 10000 7798
MC estimate for PI = 3.119200
```

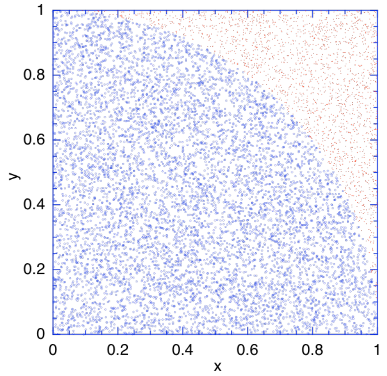


Figure: 10,000 randomly generated points, (x, y), in $[0, 1]^2$. The 7,874 blue points satisfy $x^2 + y^2 < 1$, while the remaining 2,126 (= 10,000 - 7,874) red points do not.

- **Random-number generator in the standard library**: The Unix standard library on bit includes a random-number generator, `int rand()`, note the header file, `#include <stdlib.h>`. Successive calls to this routine generate a sequence of random numbers in the range, $[0, 2^{31} - 1 = 2{,}147{,}483{,}647]$.

- `void srand( (unsigned) time((int *)0) );`
  To initialize the sequence of random numbers, use `void srand(unsigned seed)`, where `seed` is a random starting point of a sequence. In the above, the seed depends on the time the program is run. The `int time(int *tloc)` system call returns the time since 00:00:00 GMT, January 1, 1970,

measured in seconds. If `tloc` is nonzero, the return value is also stored in the place to which `tloc` points.

- `RAND_MAX` is the maximum random number, $2^{31} - 1 = 2{,}147{,}483{,}647$, defined in `/usr/include/stdlib.h`. The expression, `rand()/(float)RAND_MAX`, generates a random number in the range, [0,1].

MC INTEGRATION 2: SAMPLE MEAN

Recall the MC integration of a one-dimensional function, *f*(*x*), in the range, [*a*, *b*],

$$\int_a^b dx f(x) \cong \sum_{n=1}^{M} \frac{b-a}{M} f(r_n).$$

Let's apply this method to $f(x) = 4/(1 + x^2)$ in the range, [0, 1]. Note that

$$\int_0^1 dx \frac{4}{1+x^2} = \int_0^{\pi/4} \frac{d\theta}{\cos^2\theta} \frac{4}{1+\tan^2\theta} = \int_0^{\pi/4} 4\, d\theta = \pi .$$

(We have changed the variable from *x* to θ where $x = \tan\,\theta$.)

- Program: `mean.c`

```
/* Monte Carlo integration of PI by sample mean */
#include <stdio.h>
#include <stdlib.h>
main() {
  double x, pi, sum = 0.0;
  int try, ntry;
  printf("Input the number of MC trials\n");
  scanf("%d",&ntry);
  srand((unsigned)time((int *)0));
  for (try=0; try<ntry; try++) {
    x = rand()/(float)RAND_MAX;
    sum += 4.0/(1.0 + x*x);
  }
  pi = sum/ntry;
  printf("MC estimate for PI = %f\n", pi);
}
```

```
> cc mean.c
> a.out
Input the number of MC trials
100
MC estimate for PI = 3.127413
```

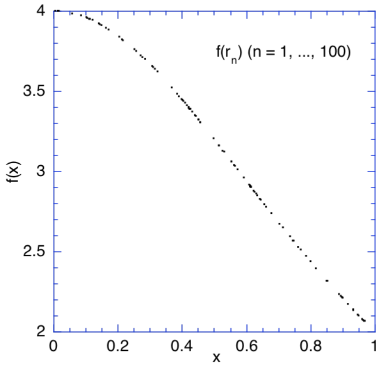


Figure: Function values, $f(x) = 4/(1 + x^2)$ for 100 random points, $x = r_n$ ($n = 1, ..., 100$).

APPLICATION: STATISTICAL MECHANICS

Consider a cubic box of length, *L*, which contains *N* particles. Statistical mechanics deals with the long-time average behavior of such a system.

Let's specify a configuration of the system using a set of coordinates, $\{\vec{r}_i = (x_i, y_i, z_i) \mid 0 < x_i, y_i, z_i < L; i = 1, ..., N\}$. The long-time average of any physical quantity, $A(\{\vec{r}_i\})$, is given by a multidimensional integral,

$$\langle A(\{\vec{r}_i\})\rangle = \frac{\int d\vec{r}_1 \cdots \int d\vec{r}_N \exp(-\frac{V(\{\vec{r}_i\})}{k_B T}) A(\{\vec{r}_i\})}{\int d\vec{r}_1 \cdots \int d\vec{r}_N \exp(-\frac{V(\{\vec{r}_i\})}{k_B T})}.$$

Here T is the temperature in Kelvin, $k_B = 1.38\times10^{-16}$ erg/K is the Boltzmann constant, and $V(\{\vec{r}_i\})$ is the potential energy of the system in erg. (An example of $A(\{\vec{r}_i\})$ is the average distance of a pair of particles.)

For example, if the particles have a charge, e, in the electrostatic unit, then the potential energy is given by

$$V(\{\vec{r}_i\}) = \sum_{i<j} \frac{e^2}{|r_i - r_j|}.$$

State-of-the-art particle simulations involve 1 billion particles, for which the above expression contains 3-billion dimensional integrals. Numerical integration using regular mesh points is prohibitive for such large systems. The simplest MC integration would generate a random point in the 3 billion-dimensional space by generating 3 billion random numbers in [0, L], and then evaluate the function at the point. In many cases, however, the factor $\exp(-V(\{\vec{r}_i\})/k_B T)$ (called the Boltzmann factor) is extremely small in most of the points except for the neighbor of the point where $V(\{\vec{r}_i\})$ takes its minimum value.

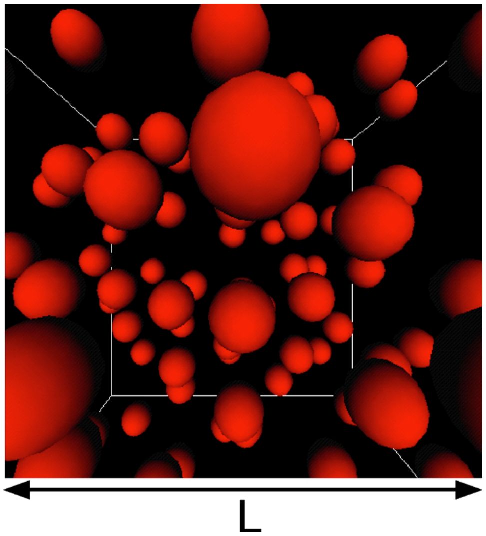


Figure: A set of N particles in a cubic box of length L.

To illustrate this, let's consider a single particle in one-dimensional space of length 1 cm. The particle is subjected to a potential, $V(x) = Ck_B(x - 0.5)^2$, where $C = 1$ K/cm^2. The figure shows the factor $\exp(-V(\{\vec{r}_i\})/k_B T)$ for temperatures 0.1, 0.01, and 0.001 K. For the lowest temperature, 0.001 K, the ranges, [0, 0.4] and [0.6, 1], contribute negligibly to the integration. By generating random numbers uniformly in [0, 1], therefore, we are wasting 80 % of the computing time. If there are 100 particles in the same 1-dimensional space all subjected to the same potential, only $0.2^{100} = 1.27\times10^{-70}$ of the total random points contribute to the integral. Your random numbers make a contribution once in every 10^{70} attempts!

An efficient MC method should generate random points preferentially near the potential minimum, $x = 0.5$. Techniques to sample important points preferentially are called **importance sampling**, which is essential for successful MC applications.

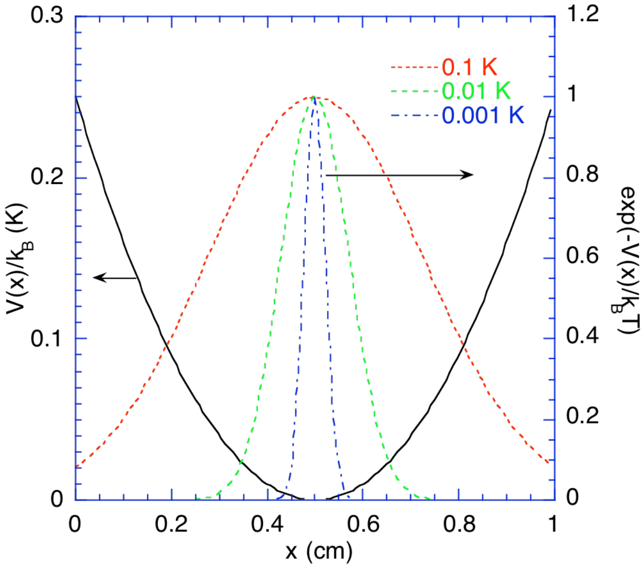


Figure: Boltzmann factor for a particle in one-dimensional factor.

§2 Probability—Foundation of the MC Method

ELEMENTS OF PROBABILITY THEORY[2]

- **Random variable**: Arises when repeated attempts under apparently identical conditions to measure a quantity produce different results. For example, the reading from tossing a dice, x ($\in \{1, 2, 3, 4, 5, 6\}$), is a random variable.
- **Probability density function**: Defines the probability, $p(x)$, when the variable is sampled, that the value x will result. For a large number of samples,

  $p(x)$ = (Number of samples with result x)/(Total number of samples)

  For example, the probability that the reading of dice is 1, $p(1)$, is 1/6.
- **Normalization condition**: Satisfied by the probability density function simply states that the number of samples for all the readings add up to the total number of samples.

  $$\sum_k p(x_k) = 1,$$

  i.e., Σ_x(Number of samples with result, x)/(Total number of samples) = 1. For example, for a dice, $p(1) + p(2) + \cdots + p(6) = 1$.
- **Expectation (mean)**: of a random variable, x, is defined as

  $$E[x] = \langle x \rangle = \sum_k x_k p(x_k).$$

  For example, the expected reading of a dice is,

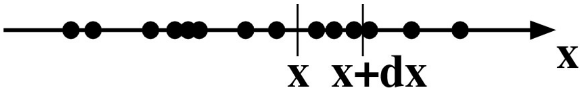
  $$1 \times \frac{1}{6} + \cdots 6 \times \frac{1}{6} = \frac{6 \times (1+6)}{2} \times \frac{1}{6} = 3.5.$$

CONTINUOUS RANDOM VARIABLES

Random variables can be continuous, e.g., the position of an atom along x axis in a box. The probability density function now defines the probability that, when the variable is sampled, a value lying in the range x to $x + dx$ will result; this probability is

$$p(x)dx.$$

This is realized in the limit of very large number of sample points such that $p(x)dx$ = (Number of samples in the range, $[x, x+dx]$)/(Total number of samples).



The normalization condition now involves an integral,

[2] M. H. Kalos and P. A. Whitlock, *Monte Carlo Methods, Vol. I: Basics* (John Wiley & Sons, New York, 1986)—a good introduction to the general MC method; P. K. MacKeown, *Stochastic Simulation in Physics* (Springer, Singapore, 1997)—contains a practical introduction to probability with many applications in physics; W. Feller, *An Introduction to Probability Theory and Its Application, Vol. I, 3rd Ed.* (John Wiley & Sons, New York, 1968)—a classic of probability theory with rigorous treatment of Markov chains.

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

and the mean is calculated as

$$E[x] = \langle x \rangle = \int_{-\infty}^{\infty} dxxp(x) .$$

We can also define the expectation of a function value,

$$E[f(x)] = \langle f(x) \rangle = \int_{-\infty}^{\infty} dxf(x)p(x) ,$$

i.e., if we repeatedly generate random numbers, x, and sample the function value, $f(x)$, then $\langle f(x) \rangle$ is the average value over the entire samples.

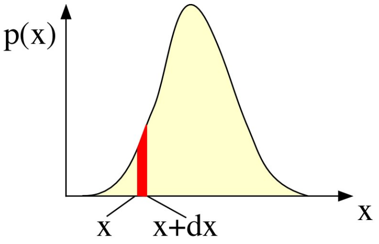


Figure: A continuous probability density function.

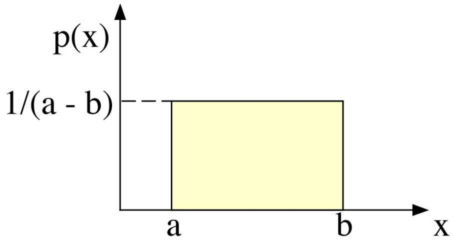


Figure: Uniform probability in the range [a, b].

For example, if a random variable, x, is uniformly distributed in the range, [a, b], then the probability density function is,

$$p(x) = 1/(a - b).$$

The mean of the random variable, x, is

$$\langle x \rangle = \int_a^b dxx \frac{1}{b-a} = \frac{1}{b-a} \left[\frac{x^2}{2} \right]_a^b = \frac{b^2 - a^2}{2(b-a)} = \frac{b+a}{2} .$$

For a function, $f(x) = x^2$, the expected value is given by

$$\langle x^2 \rangle = \int_a^b dxx^2 \frac{1}{b-a} = \frac{1}{b-a} \left[\frac{x^3}{3} \right]_a^b = \frac{b^3 - a^3}{3(b-a)} = \frac{b^2 + ab + a^2}{3} .$$

VARIANCE AND STANDARD DEVIATION

- **Variance** of a random variable, x, is defined as

$$Var[x] = \left\langle \left(x - \langle x \rangle \right)^2 \right\rangle = \left\langle x^2 - 2x\langle x \rangle + \langle x \rangle^2 \right\rangle = \langle x^2 \rangle - \langle x \rangle^2 .$$

For the above uniform distribution, the variance is

$$Var[x] = \frac{b^2 + ba + a^2}{3} - \left(\frac{b+a}{2} \right)^2 = \frac{(b-a)^2}{12} .$$

- **Standard deviation (standard error)** of a random variable, x, is the square root of its variance,

$$Std[x] = \sqrt{Var[x]} ,$$

  and it measures how broadly the probability density spreads around its mean. In the above example, the standard deviation is $|b-a|/2\sqrt{3}$.

§3 MC Estimation

SAMPLE MEAN MC

The sample mean MC estimation of a one-dimensional integral is derived as follows. Let $p(x)$ be some probability density function. (We assume that $p(x)$ is nonzero in the range, $[a, b]$.) Then rewrite the integration as

$$F = \int_a^b dx f(x) = \int_a^b dx \left(\frac{f(x)}{p(x)} \right) p(x) = \left\langle \frac{f(x)}{p(x)} \right\rangle .$$

(Recall that $E[f(x)] = \int dx p(x) f(x)$.) Thus the deterministic integration of a function is replaced by the mean value, $\langle f(x) \rangle$, of the function value over stochastic samples.

Consider a uniform distribution, $p(x) = 1/(b - a)$, then

$$\int_a^b dx f(x) = (b-a) \langle f(x) \rangle ,$$

i.e., generate a uniformly distributed random number sequence, $x = r_n$ ($n = 1, ..., M$), in the range, $[a, b]$, and calculate the mean, $\langle f(x) \rangle$, of the function value. The MC estimate of the integral is

$$\int_a^b dx f(x) = (b-a) \langle f(x) \rangle = (b-a) \times \frac{1}{M} \sum_{n=1}^{M} f(r_n) ,$$

and this is what we have programmed in `mean.c` in the previous section.

Example: Statistical mechanics

$$\langle A(\{\vec{r}_i\}) \rangle = \int d\vec{r}_1 \cdots \int d\vec{r}_N P(\{\vec{r}_i\}) A(\{\vec{r}_i\}) ,$$

where

$$P(\{\vec{r}_i\}) = \frac{\exp(-\frac{V(\{\vec{r}_i\})}{k_B T})}{\int d\vec{r}_1 \cdots \int d\vec{r}_N \exp(-\frac{V(\{\vec{r}_i\})}{k_B T})} .$$

ESTIMATION BY FINITE SAMPLING

The above argument is based on the probability density function, $p(x)$, which is defined as the limit of a large number of samples. In reality, we estimate the mean values from finite samples. The question is what is the error associated with the estimation? We now distinguish the mean in the infinite population,

$$\langle f(x) \rangle = \int dx f(x) p(x) ,$$

from that in a finite population,

$$\overline{f(x)} = \frac{1}{M} \sum_{n=1}^{M} f(r_n) .$$

Here we assume that the random numbers, $\{r_n \mid n = 1, ..., M\}$, are sampled according to the underlying probability density function. Note that a finite sample itself is a random variable (repeated M-samplings will produce different results), and accordingly has the mean and the standard deviation.

- **Unbiased estimator**: If an estimate of a statistical quantity using M samples converges to the underlying population value, the estimator is unbiased.

(Theorem) The mean of M sampled random variables, $\overline{f(x)}$, converges to its expected value, $\langle f(x) \rangle$.

$$\because E\left\{\overline{f(x)}\right\} = \frac{1}{M}\sum_{n=1}^{M} E\{f(r_n)\} = \frac{1}{M}\sum_{n=1}^{M} \langle f(x) \rangle = \frac{1}{M} M \langle f(x) \rangle = \langle f(x) \rangle .$$

The second equality holds since each sample is independent and follows the underlying probability density function.

VARIANCE OF SAMPLE MEAN

Variance of an MC estimate, $\overline{f(x)}$, of the mean is related to the standard deviation of the underlying population.

$$Var\{\overline{f(x)}\} = Var\left\{\frac{1}{M}\sum_{n=1}^{M} f(r_n)\right\} = \frac{1}{M^2}\sum_{m=1}^{M}\ \sum_{n=1}^{M}\Big\langle\big[f(r_m)-\langle f(x)\rangle\big]\big[f(r_n)-\langle f(x)\rangle\big]\Big\rangle.$$

Now an important assumption is that each sample is independent of each other, i.e., there is no correlation between any two samples. Mathematically the lack of correlation between two random variable, A and B, is expressed as $\langle AB\rangle = \langle A\rangle\langle B\rangle$. In the above equation,

$$\Big\langle\big[f(r_m)-\langle f(x)\rangle\big]\big[f(r_n)-\langle f(x)\rangle\big]\Big\rangle = \left\{\begin{array}{ll} Var\{f(x)\} & (m=n) \\ \langle f(r_m)-\langle f(x)\rangle\rangle\langle f(r_n)-\langle f(x)\rangle\rangle = 0 & \text{(else)} \end{array}\right. .$$

Substituting this relation to the above,

$$Var\{\overline{f(x)}\} = \frac{1}{M^2}\sum_{m=1}^{M} Var\{f(x)\} = \frac{Var\{f(x)\}}{M} .$$

By taking the square root,

$$Std\{\overline{f(x)}\} = \frac{Std\{f(x)\}}{\sqrt{M}} .$$

(Bottom line) The error of an MC estimation decreases as $1/\sqrt{M}$.

UNBIASED ESTIMATOR FOR THE STANDARD DEVIATION

The above equation expresses the standard deviation of the mean of finite samples in terms of that of infinite samples. The problem is that we do not know the latter. Fortunately there is an estimator for the standard deviation of the underlying infinite population from finite samples.
Let's define the variance, s_M, measured in M samples,

$$s_M = \frac{1}{M}\sum_{n=1}^{M} f^2(r_n) - \left[\frac{1}{M}\sum_{n=1}^{M} f(r_n)\right]^2 .$$

Note again that s_M itself is a random variable. The value may vary from time to time, but its expectation value is

$$\begin{aligned}\langle s_M\rangle &= \frac{1}{M}\sum_{n=1}^{M}\langle f^2(x)\rangle - \frac{1}{M^2}\sum_{m=1}^{M}\sum_{n=1}^{M}\langle f(r_m)f(r_n)\rangle \\ &= \langle f^2(x)\rangle - \frac{1}{M^2}\sum_{m=1}^{M}\langle f^2(x)\rangle - \frac{1}{M^2}\sum_{m=1}^{M}\sum_{n=1(n\neq m)}^{M}\langle f(r_m)f(r_n)\rangle.\end{aligned}$$

For uncorrelated random numbers,

$$\langle f(r_m)f(r_n)\rangle = \langle f(x)\rangle\langle f(x)\rangle,$$

and therefore

$$\begin{aligned}\langle s_M\rangle &= \left(1-\frac{1}{M}\right)\langle f^2(x)\rangle - \frac{M(M-1)}{M^2}\langle f(x)\rangle\langle f(x)\rangle \\ &= \left(1-\frac{1}{M}\right)Var\{f(x)\}.\end{aligned}$$

For the large *M* limit,

$$\frac{M}{M-1}s_M = \frac{M}{M-1}\left\{\frac{1}{M}\sum_{n=1}^{M}f^2(r_n) - \left[\frac{1}{M}\sum_{n=1}^{M}f(r_n)\right]^2\right\} \to \quad Var\{f(x)\} \quad (M\to\infty)$$

converges to *Var*{*f*(*x*)} and thus it is an unbiased estimator of the latter.

VARIANCE OF SAMPLE MEAN IN PRACTICE

We can now use the relation

$$Var\{\overline{f(x)}\} = \frac{Var\{f(x)\}}{M},$$

to estimate the variance of the MC mean (i.e., what is the uncertainty associated with the MC sample mean), since we now have an estimator for *Var*{*f*(*x*)},

$$\frac{M}{M-1}s_M = \frac{M}{M-1}\left\{\frac{1}{M}\sum_{n=1}^{M}f^2(r_n) - \left[\frac{1}{M}\sum_{n=1}^{M}f(r_n)\right]^2\right\} \to \quad Var\{f(x)\} \quad (M\to\infty) \quad .$$

Combining these two equations, we have

$$Var\{\overline{f(x)}\} = \frac{1}{M-1}s_M = \frac{1}{M-1}\left\{\frac{1}{M}\sum_{n=1}^{M}f^2(r_n) - \left[\frac{1}{M}\sum_{n=1}^{M}f(r_n)\right]^2\right\} \to \quad \frac{Var\{f(x)\}}{M} \quad (M\to\infty) \quad .$$

Let us summarize the main results on the sample mean MC method.

Mean:

$$\overline{f(x)} = \frac{1}{M}\sum_{n=1}^{M} f(r_n) \to \ \langle f(x)\rangle = \int dx f(x) p(x) \quad (M \to \infty) \quad .$$

Variance of the mean:

$$Var\{\overline{f(x)}\} = \frac{1}{M-1} s_M = \frac{1}{M-1}\left\{\frac{1}{M}\sum_{n=1}^{M} f^2(r_n) - \left[\frac{1}{M}\sum_{n=1}^{M} f(r_n)\right]^2\right\} \to \ \frac{Var\{f(x)\}}{M} \quad (M \to \infty) \quad .$$

CENTRAL THEOREM ON MC SAMPLING

$$\int dx f(x) p(x) \cong \bar{f} \pm \sqrt{\frac{\overline{f^2} - \left(\bar{f}\right)^2}{M-1}} \ ,$$

where finite-samplings are defined through random variables, r_n, according to $p(x)$, as

$$\bar{A} \equiv \frac{1}{M}\sum_{n=1}^{M} A\left(r_n\right) .$$

§4 Importance Sampling

In the previous section, we have learned how we can estimate the mean value of a random function and its standard error both from random sampling. The question now is how to sample random variables according to the underlying probability density, $p(x)$. For example, recall the statistical mechanics in the previous section. The statistical average of a function $A(\vec{r}^N)$ (where $\vec{r}^N = \{\vec{r}_1, \vec{r}_2, ..., \vec{r}_N\}$) is estimated by generating random samples, $\{\vec{r}_1^N, \vec{r}_2^N, ..., \vec{r}_M^N\}$, as

$$\langle A \rangle = \frac{\int d\vec{r}^N A(\vec{r}^N) \exp\left(-V(\vec{r}^N)/k_BT\right)}{\int d\vec{r}^N \exp\left(-V(\vec{r}^N)/k_BT\right)} \approx \frac{\sum_{n=1}^{M} A(\vec{r}_n^N) \exp\left(-V(\vec{r}_n^N)/k_BT\right)}{\sum_{n=1}^{M} \exp\left(-V(\vec{r}_n^N)/k_BT\right)}.$$

However, we have learned that this approach is practically useless and we instead need to generate random points according to the probability that is proportional to $\exp(-V(\vec{r}^N)/k_BT)$. This is called importance sampling.

IMPORTANCE SAMPLING

The discussion of the sample mean MC method in the previous section has assumed that we can generate a sequence of random points, r_1, r_2, ..., so that their population follows a given probability density function, $p(x)$.

- **Importance sampling**: Chooses random numbers from a probability density function, $p(x)$.

One importance sampling method generates a *sequence* of random points so that the population of the entire points follows the given probability density rather than generating each point with the probability density (the latter is more difficult).

MARKOV CHAIN

For simplicity, let's consider a finite-state system which at any moment can be in one of the states, $\{\Gamma_1, \Gamma_2, ..., \Gamma_N\}$. An example is a system of n atoms in a cubic box in which the coordinate in each direction is discretized so that it takes only one of m discretized grid points. A state of the system is specified by $3n$ discretized points, and there are m^{3n} combinations for possible states. The probability density function in this case is a $3n$-dimensional function, $p(\Gamma_1, \Gamma_2, ..., \Gamma_N)$.

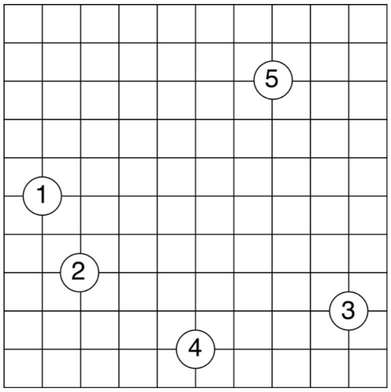


Figure: A two-dimensional system with 5 atoms discretized on 11 grid points in each direction. The number of all possible combinations of atomic positions is 11^{10}.

- **Markov chain**: A sequence of trials that satisfies two conditions:
  (a) The outcome of each trial belongs to a finite set of outcomes, $\{\Gamma_1, \Gamma_2, ..., \Gamma_N\}$, called the state space;
  (b) The outcome of each trial depends only on the outcome of the trial that immediately precedes it (the system has a very short memory).
- **Transition probability matrix**: Any two states Γ_n and Γ_m are linked by a transition probability, π_{mn}, which is the *conditional probability* that the next state is Γ_m given that the current state Γ_n, i.e., the

probability of state transition, $n \to m$. Each transition probability π_{mn} is an element of the transition probability matrix,

$$\Pi = \begin{bmatrix} \pi_{11} & \pi_{12} & \cdots & \pi_{1N} \\ \pi_{21} & \pi_{22} & \ddots & \\ \vdots & & & \\ \pi_{N1} & & & \pi_{NN} \end{bmatrix}.$$

- **Normalization**: Given that the system is in the state, Γ_n, then the next state must be one of the N possible states so that

$$\sum_{m=1}^{N} \pi_{mn} = 1 .$$

Suppose we start (with 100% probability) from one of the states, say n, then the next state is stochastic with probability,

$$\rho^{(1)} = \begin{pmatrix} p_1 \\ p_2 \\ \vdots \\ p_N \end{pmatrix} = \Pi \rho^{(0)} = \Pi \begin{pmatrix} 0 \\ \vdots \\ 1 \\ \vdots \end{pmatrix} \begin{matrix} \\ \\ n \\ \\ \end{matrix} \quad .$$

($p_m^{(1)} = \sum_{n'=1}^{N} \pi_{mn'} \delta_{n'n} = \pi_{mn}$, where $\delta_{nn'} = 1$ if $n = n'$ and 0 otherwise.)

The probability distribution of the τ-th time step is given by successive matrix multiplications:

$$\rho^{(\tau)} = \Pi \rho^{(\tau-1)} = \Pi^2 \rho^{(\tau-2)} = \cdots = \Pi^\tau \rho^{(0)} .$$

The following theorem[3] gives us a clue how a Markov chain is used to achieve a population with a given probability density.

- **Perron-Frobenius theorem**: A transition probability matrix has one eigenvalue that equals unity, and the corresponding eigenvector is the limiting distribution of the chain. The absolute values of all the other eigenvalues are less than unity.

1. The consequence of the Perron-Frobenius theorem is that Π^τ for large τ converges to a matrix which, acting on any initial probability distribution, $\rho^{(0)}$, converges to the eigenvector of the unit eigenvalue. All the other eigenvectors, whose eigenvalue being less than unity, are filtered out by successive matrix multiplications. ($\rho^\infty = \lim_{\tau \to \infty} \Pi^\tau \rho^{(0)}$.)
2. In order to achieve a distribution of states, which follows a given probability density, find a transition probability matrix whose eigenvector with unit eigenvalue (the existence of which is guaranteed by the above normalization condition) equals the desired probability density.

(Example—Two State System)[4]

Suppose the reliability of a (rather unreliable) computer follows a certain pattern. If it is up and running on one day, it has a 60% of running correctly on the next. If it is down, it has a 70% chance of also being down the next day.

[3] For a proof, see W. Feller, *Introduction to Probability Theory and Its Applications, Vol. I* (John Wiley & Sons, New York, 1968).

[4] M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids* (Oxford Univ. Press, Oxford, 1987) p. 115.

The transition probability matrix of this computer is given by,

$$\Pi = \begin{array}{c c} & \begin{array}{cc} \uparrow & \downarrow \end{array} \\ \begin{array}{c} \uparrow \\ \downarrow \end{array} & \begin{pmatrix} a & 1-b \\ 1-a & b \end{pmatrix} \end{array},$$

where $a = 0.6$ and $b = 0.7$. For example, if the computer is up today,

$$\begin{pmatrix} p_{\uparrow}^{(1)} \\ p_{\downarrow}^{(1)} \end{pmatrix} = \begin{pmatrix} 0.6 & 0.3 \\ 0.4 & 0.7 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix},$$

i.e., it will be 60% up and 40 % down tomorrow. For the day after tomorrow,

$$\begin{pmatrix} p_{\uparrow}^{(2)} \\ p_{\downarrow}^{(2)} \end{pmatrix} = \begin{pmatrix} 0.6 & 0.3 \\ 0.4 & 0.7 \end{pmatrix} \begin{pmatrix} 0.6 \\ 0.4 \end{pmatrix} = \begin{pmatrix} 0.48 \\ 0.52 \end{pmatrix}.$$

In order to obtain a closed form for the probability on day n, we need to solve an eigenvalue problem,

$$\Pi w = \lambda w,$$

where λ is an eigenvalue and the corresponding eigenvector is

$$w = \begin{pmatrix} u \\ v \end{pmatrix}.$$

We can rewrite the eigenvalue equation as

$$\begin{pmatrix} a & 1-b \\ 1-a & b \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \lambda \begin{pmatrix} u \\ v \end{pmatrix},$$

or

$$\begin{pmatrix} \lambda - a & b-1 \\ a-1 & \lambda - b \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

For the above equation to have a nontrivial solution (trivial solution is $u = v = 0$), there should not exist the inverse matrix of Π, or the determinant of the matrix must be zero,

$$|\Pi| = (\lambda - a)(\lambda - b) - (a-1)(b-1) = (\lambda - a - b + 1)(\lambda - 1) = 0.$$

There are two solutions, $\lambda_+ = 1$ and $-1 < \lambda_- = a+b-1 < 1$. (For each eigenvalue, the two equations constituting the matrix equation become identical: $(1-a)u_+ + (b-1)v_+ = 0$ for λ_+ and $u_- + v_- = 0$ for λ_-.) The corresponding eigenvectors are

$$w_+ = \begin{pmatrix} u_+ \\ v_+ \end{pmatrix} = \frac{1}{2-a-b} \begin{pmatrix} 1-b \\ 1-a \end{pmatrix} \quad \text{and} \quad w_- = \begin{pmatrix} u_- \\ v_- \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}.$$

(Normalization of the vector is not important.) We can combine the two eigensolutions in a matrix equation,

$$\Pi U = U \Lambda,$$

where U is called the transformation matrix,

$$U = \begin{pmatrix} u_+ & u_- \\ v_+ & v_- \end{pmatrix} = \begin{pmatrix} (1-b)/(2-a-b) & -1 \\ (1-a)/(2-a-b) & 1 \end{pmatrix} \quad \text{and} \quad \Lambda = \begin{pmatrix} \lambda_+ & 0 \\ 0 & \lambda_- \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & a+b-1 \end{pmatrix}.$$

$\because$

$$\begin{pmatrix} a & 1-b \\ 1-a & b \end{pmatrix}\begin{pmatrix} u_+ \\ v_+ \end{pmatrix} = \lambda_+ \begin{pmatrix} u_+ \\ v_+ \end{pmatrix} = \begin{pmatrix} \lambda_+ u_+ \\ \lambda_+ v_+ \end{pmatrix}$$

$$\begin{pmatrix} a & 1-b \\ 1-a & b \end{pmatrix}\begin{pmatrix} u_- \\ v_- \end{pmatrix} = \lambda_- \begin{pmatrix} u_- \\ v_- \end{pmatrix} = \begin{pmatrix} \lambda_- u_- \\ \lambda_- v_- \end{pmatrix}$$

By combining these two matrix equations,

$$\begin{pmatrix} a & 1-b \\ 1-a & b \end{pmatrix}\begin{pmatrix} u_+ & u_- \\ v_+ & v_- \end{pmatrix} = \begin{pmatrix} \lambda_+ u_+ & \lambda_- u_- \\ \lambda_+ v_+ & \lambda_- v_- \end{pmatrix} = \begin{pmatrix} u_+ & u_- \\ v_+ & v_- \end{pmatrix}\begin{pmatrix} \lambda_+ & 0 \\ 0 & \lambda_- \end{pmatrix} //$$

Or, by multiplying the inverse of U from the right on both sides of the above matrix equation,

$$\Pi = U\Lambda U^{-1},$$

where the inverse of the transformation matrix is

$$U^{-1} = \begin{pmatrix} 1 & 1 \\ (a-1)/(2-a-b) & (1-b)/(2-a-b) \end{pmatrix}.$$

Let's multiply the transition probability matrix repeatedly to an arbitrary initial probability distribution,

$$\begin{pmatrix} p_1^{(n)} \\ p_2^{(n)} \end{pmatrix} = \Pi^n \begin{pmatrix} p_1 \\ p_2 \end{pmatrix}.$$

Meaning of such a vector is as follows. Starting from the same initial condition, we monitor the time sequence of up and down state for many samples. On day τ, count the number of up's and down's in the ensemble, and these give the up and down probabilities on the day.

$$\begin{aligned} \begin{pmatrix} p_1^{(n)} \\ p_2^{(n)} \end{pmatrix} &= \Pi^n \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \\ &= U\Lambda^n U^{-1} \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \\ &= \begin{pmatrix} u_+ & -1 \\ v_+ & 1 \end{pmatrix}\begin{pmatrix} \lambda_+^n & 0 \\ 0 & \lambda_-^n \end{pmatrix}\begin{pmatrix} 1 & 1 \\ -v_+ & u_+ \end{pmatrix}\begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \\ &= \begin{pmatrix} u_+ & -1 \\ v_+ & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} 1 & 1 \\ -v_+ & u_+ \end{pmatrix}\begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \quad (n \to \infty) \\ &= \begin{pmatrix} u_+(p_1+p_2) \\ v_+(p_1+p_2) \end{pmatrix} = \begin{pmatrix} u_+ \\ v_+ \end{pmatrix}. \end{aligned}$$

Note that in the above derivation,

$$U\Lambda U^{-1}U\Lambda U^{-1}\cdots U\Lambda U^{-1} = U\Lambda\left(U^{-1}U\right)\Lambda\cdots\Lambda\left(U^{-1}U\right)\Lambda U^{-1} = U\Lambda^n U^{-1}.$$

Therefore repeated application of the transition probability matrix to an arbitrary initial probability density converges to the eigenvector corresponding to the unit eigenvalue, and this proves the Perron-Frobenius theorem for an arbitrary two-state system. Note that we have used the fact,

$$\lambda_-^n \to 0 \quad (n \to \infty, \left|\lambda_-^n\right| < 1) \quad .$$

Eventually, the computer has a $u_+ = (1 - b)/(2 - a - b) = 0.3/0.7 = 0.4286$ chance of being up and running at any day, and $u_- = (1 - a)/(2 - a - b) = 0.4/0.7 = 0.5714$ chance of being down.

METROPOLIS ALGORITHM[5]

Now the question of generating a desired probability distribution has reduced to finding a transition probability matrix whose eigenvector with unit eigenvalue equals to the probability density function. To do this, recall the eigenvalue equation with unit eigenvalue,

$$\Pi\rho = \rho \quad \text{or} \quad \sum_{n=1}^{N} \Pi_{mn}\rho_n = \rho_m .$$

The problem is that given a desired ρ, how to find Π which satisfies the above relation. A trick is to impose the stronger condition below.

- **Detailed balance condition**: For any pair of states,

$$\pi_{mn}\rho_n = \pi_{nm}\rho_m .$$

  The left-hand side is a *flux* of probability that the current state is n and that the next state is m. The right-hand side is a flux from m to n. For any pair of states this balance of fluxes must be microscopically balanced.

In order to show that above detailed balance condition is sufficient for the unit-eigenvalue relation, sum the both side over n.

$$\because \sum_{n=1}^{N} \pi_{mn}\rho_n = \underbrace{\sum_{n=1}^{N} (\pi_{nm})}_{1} \rho_m$$

$$\therefore \sum_{n=1}^{N} \pi_{mn}\rho_n = \rho_m .$$

- **Metropolis' transition probability matrix**: Called an asymmetric solution uses a symmetrical ($\alpha_{mn} = \alpha_{nm}$) *attempt* matrix:

$$\pi_{mn} = \begin{cases} \alpha_{mn} & \rho_m \ge \rho_n \quad m \ne n \\ (\rho_m/\rho_n)\alpha_{mn} & \rho_m < \rho_n \quad m \ne n \\ 1 - \sum_{m' \ne n} \pi_{m'n} & \qquad\qquad m = n \end{cases} .$$

The Metropolis's method is a kind of *rejection* method—propose and accept/reject: It first attempts a move from $n \to m$ with probability α_{mn}. If the destination probability is larger than the source probability, then unconditionally *accept* the attempt. Otherwise if the destination probability is smaller than the source, accept the attempt with a probability, ρ_m/ρ_n. The last relation guarantees the normalization condition on the transition probability matrix.

Proof of detailed balance:
Assume that $\rho_m < \rho_n$ (similar proof for the other case). $\pi_{nm}\rho_m = \alpha_{nm}\rho_m = \alpha_{mn}\rho_m$ due to the symmetrical attempt. On the other hand, $\pi_{mn}\rho_n = (\rho_m/\rho_n)\alpha_{mn}\rho_n = \alpha_{mn}\rho_m$ .//

[5] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, Equation of states calculations by fast computing machines, J. Chem. Phys. **21**, 1087 (1953).

An advantage of the Metropolis' algorithm is that the relative probability density ρ_m/ρ_n is relatively easy to calculate compared with the absolute probability ρ_n. For example, let's consider the probability density in statistical mechanics,

$$P(\{\vec{r}_i\}) = \frac{\exp(-\frac{V(\{\vec{r}_i\})}{k_BT})}{\int d\vec{r}_1 \cdots \int d\vec{r}_N \exp(-\frac{V(\{\vec{r}_i\})}{k_BT})},$$

which is known only after we integrate the exponential factor over the entire configuration space (an intractable task). The Metropolis' method only requires the evaluation of a relative probability,

$$\frac{P\left(\left\{\vec{r}_i^{\,next}\right\}\right)}{P\left(\left\{\vec{r}_i^{\,current}\right\}\right)} = \frac{\exp\left(-V\left(\left\{\vec{r}_i^{\,next}\right\}\right)/k_BT\right)}{\exp\left(-V\left(\left\{\vec{r}_i^{\,current}\right\}\right)/k_BT\right)} = \exp\left(-\delta V\left(\left\{\vec{r}_i^{\,next}\right\},\left\{\vec{r}_i^{\,current}\right\}\right)/k_BT\right),$$

where

$$\delta V\left(\left\{\vec{r}_i^{\,next}\right\},\left\{\vec{r}_i^{\,current}\right\}\right) = V\left(\left\{\vec{r}_i^{\,next}\right\}\right) - V\left(\left\{\vec{r}_i^{\,current}\right\}\right).$$

Note the denominator of the probability density function (or the normalization factor) cancels out. Now the only requirement is that we are able to calculate the change in the potential energy after and before each attempted change in configuration.

* Caution: The system remains in the same state if an attempt fails according to the third relation. If you are calculating the statistical average of some quantity, this same next state must be taken into account in the averaging process again in addition to the already counted current state. The following pseudocode demonstrates this point.

- Template: Metropolis algorithm for statistical mechanics

```
Sum_A = 0
for n = 1 to Ntrial
  X′ ← X + dX
  Compute dV = V(X′) - V(X)
  if dV < 0 then  /* Greedy */
    Accept X ← X′
  else if rand() < exp(-dV/k_BT) then /* Uphill move accepted with a probability*/
    Accept X ← X′
  /* else the state remains X */
  endif
  Sum_A = Sum_A + A(X)        /* Counting X more than once if a move rejected */
endfor
Avg_A = Sum_A/Ntrial
```

* Often an attempt used in the Metropolis' algorithm is a random displacement. For example, in the discretized 3D atom system, an attempt is made with equal probability, 1/6, from 6 neighbor grid points, {north, south, east, west, up, down}.

§5 Random Number Generator

LINEAR CONGRUENTIAL GENERATOR[6]

- **Uniform random number generator**: A routine to generate a sequence of random numbers within a specified range (typically 0 to 1).

The simplest uniform random number generator uses a sequence of large positive integers X_i, each generated from the previous one by multiplication conducted in finite modulus arithmetic:

$$X_{i+1} = aX_i \bmod M,$$

where a and M are large positive integers and mod is the modulo operator. Each integer in the sequence lies in the range $[1, M-1]$. Random numbers r_i in the range [0, 1] is obtained by

$$r_i = X_i/M.$$

Since X_i can never exceed $M - 1$ the sequence must repeat after at most $M - 1$ iterations. If M is chosen to be prime and a is carefully chosen, the maximum possible period of $M - 1$ is achieved.

The most famous generator is $(M, a) = (2^{31} - 1 = 2147483647, 7^5 = 16807)$, in which M is the largest prime that can be represented by a signed 32-bit integer. This cannot be implemented using the 32-bit integer arithmetic since the multiplication can easily overflow.

The following algorithm (due to Schrage) solves this difficulty by using a factorization of m,

$$m = aq + r, \text{ i.e., } q = [m/a], r = m \bmod a$$

with square brakets denoting integer part. If $r < q$, then we can show, for an arbitrary z in the range $(0, m-1)$, that both $a(z \bmod q)$ and $r[z/q]$ lie in the range $[0, m-1]$ and

$$az \bmod m = \begin{cases} a(z \bmod q) - r[z/q] & \text{if} \geq 0 \\ a(z \bmod q) - r[z/q] + m & \text{otherwise} \end{cases}.$$

∵ Let factorize $z = xq + y$ ($y < q$: remainder) so that $x = [z/q]$ and $y = z \bmod q$. Then

$$\begin{aligned} az &= a(xq + y) \\ &= x(m - r) + ay \\ &= xm + (ay - xr). \end{aligned}$$

This suggests that $ay - xr$ is equal to $az \bmod m$ in finite modulus arithmetic. Now note that

$0 < ay < aq = m - r \leq m$, therefore $0 < ay < m$

$0 < xr \leq ar$ ($z < m$ and x & a are their respective quotients) $< aq$ (r is a remainder with divider q) $= m - r \leq m$, therefore $0 < xr < m$

Therefore $-m < ay - xr < m$, and it is equal to either ($az \bmod m$) itself or ($az \bmod m - m$). //

The following random number generator uses this algorithm with $q = [2147483647/16807] = 127773$ and $r = 2147483647 - 16807 \times 127773 = 2836$.

- The cycle of this random number generator is $m - 1 = 2^{31} - 2 \sim 2.1 \times 10^9$: You can obtain up to 2 billion independent random numbers.

[6] M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids* (Oxford Univ. Press, Oxford, 1987) Appendix G; W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C, 2nd Ed.* (Cambridge Univ. Press, Cambridge, 1992) p. 274.

- Program ran0.c

```
#define IA 16807
#define IM 2147483647
#define AM (1.0/IM)
#define IQ 127773
#define IR 2836
float ran0(int *idum) {
/*
Random number generator.  Returns a uniform random number between 0.0 and 1.0.
Set *idum (to be modified) to any nonzero integer value to initialize the sequence.
*/
  long k;
  float ans;

  k = (*idum)/IQ;
  *idum = IA*(*idum - k*IQ) - IR*k;
  if (*idum < 0) *idum += IM;
  ans = AM*(*idum);
  return ans;
}
```

The “perfect random-number generator” in *Numerical Recipes* by W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling (Cambridge Univ. Press) is an extension of this routine with “shuffle” and “mix” operations.

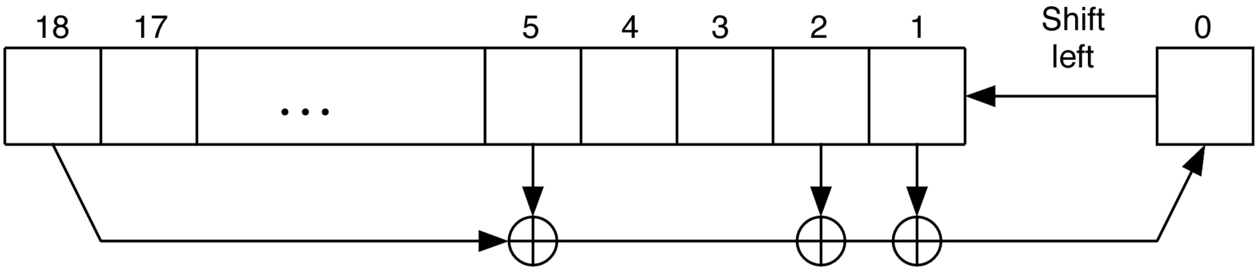
SHIFT-REGISTER (XOR) GENERATOR

We are often interested in generating single random bits, with 0 and 1 equally probable. The shift-register or Tausworthe generator uses a recurrence relation for obtaining a new random bit from the n preceding ones.[7] Let the bits be numbered from 1 (most recently generated) through n (generated n steps ago), and denoted $a_1, a_2, ..., a_n$. We want to give a formula for a new bit a_0. After generating a_0 we will shift all the bits by one, so that the old a_n is finally lost, and the new a_0 becomes a_1. We then apply the formula again, and so on.

Particularly useful recurrence relations are obtained from the theory of primitive polynomial modulo 2 of order n. One can seed the sequence with any initial bit pattern (except all zeros), and get 2^n - 1 random bits before the sequence repeats. An example of order n = 18 is

$$a_0 = a_{18} \oplus a_5 \oplus a_2 \oplus a_1 \,,$$

where the “exclusive or” (XOR) logical operator is used.



[7] R. C. Tausworthe, *Mathematics of Computation* **19**, 201 (1965).

This is suited to hardware implementation using a shift register. In the C programming language, we can use the 18 low-significant bits of unsigned integer variables and the >> (shift right) and << (shift left) operators to implement the recursion.[8]

- Program: `irbit1.c`

```
#define IB1 1  /* Bit masks */
#define IB2 2
#define IB5 16
#define IB18 131072
int irbit1(unsigned long *iseed) {
  unsigned long newbit;
  newbit = (*iseed & IB18) >> 17  /* Get bit 18 at the right-most digit */
  ^ (*iseed & IB5) >> 4           /* XOR with bit 5 */
  ^ (*iseed & IB2) >> 1
  ^ (*iseed & IB1);
  *iseed = (*iseed << 1) | newbit /* Left shift the seed & fill bit 1 with newbit
*/
  return (int) newbit;
}
```

FIBONACCI GENERATOR[9]

Constructing a lagged sequence can be done modulo $m > 2$, e.g., taking the p-th and q-th preceding numbers and combining them with some operation,

$$\nu_i = \left(\nu_{i-q} \Diamond \nu_{i-p}\right) \mod m \quad ,$$

where $\Diamond$ is the conventional bitwise XOR. Recently, however, the arithmetic subtraction has also been used. In the latter case, we add m to $\nu_{i-q} \Diamond \nu_{i-p}$ if it becomes negative.

NCSA'S SPRNG[10]

Most modern random-number generators involve the combination of two or more of elementary generators. The scalability and portability issues are addressed in some public-domain software systems such as the SPRNG at the NCSA at University of Illinois, Urbana.

[8] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C, 2nd Ed.* (Cambridge Univ. Press, Cambridge, 1992).

[9] P. K. MacKeown, *Stochastic Simulation in Physics* (Springer, Singapore, 1997) p. 76.

[10] http://www.ncsa.uiuc.edu/Apps/SPRNG

APPENDIX A—GAUSSIAN INTEGRAL

Consider the Gaussian integral,

$$I = \int_{-\infty}^{\infty} dx \exp\left(-x^2\right).$$

A common trick is instead to evaluate

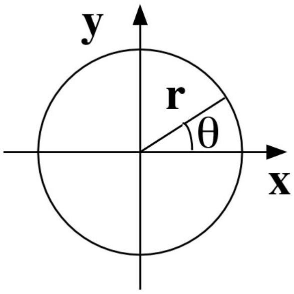
$$\begin{aligned} I^2 &= \int_{-\infty}^{\infty} dx \exp\left(-x^2\right) \int_{-\infty}^{\infty} dy \exp\left(-y^2\right) \\ &= \int_{-\infty}^{\infty} dx \int_{-\infty}^{\infty} dy \exp\left(-(x^2 + y^2)\right) \end{aligned}$$

and to introduce a polar coordinate, ($x = r \cos\theta$, $y = r \sin\theta$). Note that

$$\begin{aligned} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} dx\, dy = \int_0^{\infty} \int_0^{2\pi} dr\, d\theta \begin{vmatrix} \frac{\partial x}{\partial r} & \frac{\partial x}{\partial \theta} \\ \frac{\partial y}{\partial r} & \frac{\partial y}{\partial \theta} \end{vmatrix} &= \int_0^{\infty} \int_0^{2\pi} dr\, d\theta \begin{vmatrix} \cos\theta & -r\sin\theta \\ \sin\theta & r\cos\theta \end{vmatrix} = \int_0^{\infty} \int_0^{2\pi} dr\, d\theta \left| r\cos^2\theta + r\sin^2\theta \right| \\ &= \int_0^{\infty} \int_0^{2\pi} r\, dr\, d\theta \end{aligned}$$

(2D integration is a summation of function values multiplied by a small area, see Appendix B). Using this transformation,

$$I^2 = \int_0^{\infty} dr \int_0^{2\pi} d\theta r \exp\left(-r^2\right) = 2\pi \int_0^{\infty} dr r \exp\left(-r^2\right).$$



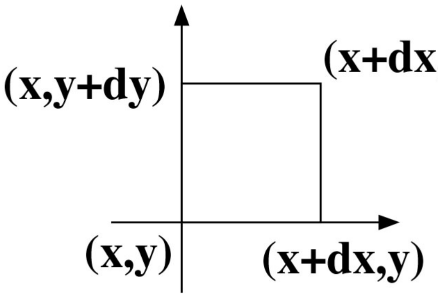
Now a further coordination transformation, $x = r^2$ and $dx = (dx/dr)dr = 2rdr$, makes

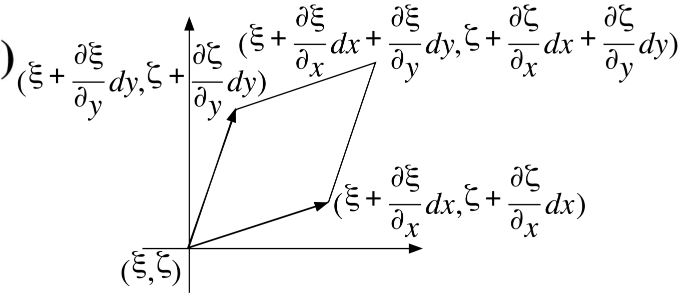
$$I^2 = \pi \int_0^{\infty} dx \exp\left(-x\right) = \pi \left[-\exp(-x)\right]_0^{\infty} = \pi .$$

Therefore, $I = \int_0^{\infty} dx \exp\left(-x^2\right) = \sqrt{\pi}$.

APPENDIX B—TRANSFORMATION OF AREAL ELEMENT

Q. Consider a variable transformation from (x, y) to $(\xi, \zeta) = (\xi(x, y), \zeta(x, y))$ and a tiny rectangle in the xy-space with area *dxdy*, which will be transformed to a parallelogram in the ξζ-space shown below. What is the area of the transformed parallelogram?




$(\xi+\frac{\partial\xi}{\partial y}dy, \zeta+\frac{\partial\zeta}{\partial y}dy)$
$(\xi+\frac{\partial\xi}{\partial x}dx+\frac{\partial\xi}{\partial y}dy, \zeta+\frac{\partial\zeta}{\partial x}dx+\frac{\partial\zeta}{\partial y}dy)$
$(\xi+\frac{\partial\xi}{\partial x}dx, \zeta+\frac{\partial\zeta}{\partial x}dx)$
(ξ,ζ)

A. $S = \begin{vmatrix} \frac{\partial\xi}{\partial x} & \frac{\partial\xi}{\partial y} \\ \frac{\partial\zeta}{\partial x} & \frac{\partial\zeta}{\partial y} \end{vmatrix} dxdy = \left(\frac{\partial\xi}{\partial x}\frac{\partial\zeta}{\partial y} - \frac{\partial\xi}{\partial y}\frac{\partial\zeta}{\partial x}\right)dxdy$, where $\begin{vmatrix} a & b \\ c & d \end{vmatrix} = |ad-bc|$ is the determinant of matrix $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$.

(Proof)

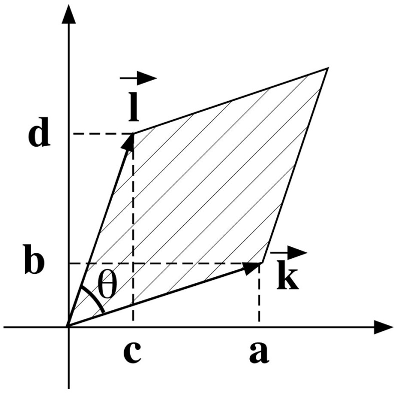
Consider a parallelogram formed by two 2D vectors $\vec{k}=(a,b)$ and $\vec{l}=(c,d)$ with the angle θ between them. Note that

$$\cos\theta = \frac{\vec{k}\bullet\vec{l}}{|\vec{k}||\vec{l}|} = \frac{ac+bd}{\sqrt{(a^2+b^2)(c^2+d^2)}}$$

$$|\sin\theta| = \sqrt{1-\cos^2\theta} = \frac{\sqrt{(a^2+b^2)(c^2+d^2)-(ac+bd)^2}}{\sqrt{(a^2+b^2)(c^2+d^2)}} = \frac{|ad-bc|}{\sqrt{(a^2+b^2)(c^2+d^2)}}$$

The area of the parallelogram is

$$S = |\vec{k}||\vec{l}|\sin\theta = \sqrt{a^2+b^2}\sqrt{c^2+d^2}\sin\theta = |ad-bc|$$



The parallelogram under question is instead formed by two vectors,

$$\vec{k} = \left(\frac{\partial\xi}{\partial x}dx, \frac{\partial\zeta}{\partial x}dx\right),$$

$$\vec{l} = \left(\frac{\partial\xi}{\partial y}dy, \frac{\partial\zeta}{\partial y}dy\right).$$

Here, we have used the linear approximation,

$$(\xi(x+dx,y+dy),\zeta(x+dx,y+dy)) \approx (\xi(x,y)+\frac{\partial\xi}{\partial x}dx+\frac{\partial\xi}{\partial y}dy,\zeta(x,y)+\frac{\partial\zeta}{\partial x}dx+\frac{\partial\zeta}{\partial y}dy)$$

With this substitution, we get

$$S=\left|\frac{\partial\xi}{\partial x}dx\frac{\partial\zeta}{\partial y}dy-\frac{\partial\zeta}{\partial x}dx\frac{\partial\xi}{\partial y}dy\right|=\left|\frac{\partial\xi}{\partial x}\frac{\partial\zeta}{\partial y}-\frac{\partial\zeta}{\partial x}\frac{\partial\xi}{\partial y}\right|dxdy=\left|\begin{array}{cc}\frac{\partial\xi}{\partial x} & \frac{\partial\xi}{\partial y}\\ \frac{\partial\zeta}{\partial x} & \frac{\partial\zeta}{\partial y}\end{array}\right|dxdy\ .$$

PROBLEM—NORMAL DISTRIBUTION

What is the standard deviation of the random number, ζ, that follow the normal probability density,

$$P(\zeta)=\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{\zeta^2}{2\sigma^2}\right)?$$

(Answer)

Note that

$$I(\sigma)=\int_{-\infty}^{\infty}d\zeta\exp\left(-\frac{\zeta^2}{2\sigma^2}\right)=\int_{-\infty}^{\infty}\sqrt{2}\sigma\,ds\exp\left(-s^2\right)=\sqrt{2\pi}\sigma\ ,$$

where we have introduced a new variable, s, through $\zeta=\sqrt{2}\sigma s$. By differentiate both sides by σ,

$$\frac{dI}{d\sigma}=\int_{-\infty}^{\infty}d\zeta\frac{\zeta^2}{\sigma^3}\exp\left(-\frac{\zeta^2}{2\sigma^2}\right)=\sqrt{2\pi}\ .$$

Or

$$\int_{-\infty}^{\infty}\frac{d\zeta}{\sqrt{2\pi}\sigma}\zeta^2\exp\left(-\frac{\zeta^2}{2\sigma^2}\right)=\left\langle\zeta^2\right\rangle=\sigma^2\ .$$

From the symmetry, the average value, $\langle\zeta\rangle = 0$, and therefore the variance of ζ is $\left\langle\zeta^2\right\rangle-\left\langle\zeta\right\rangle^2=\sigma^2$ and the standard deviation is σ.