Monte Carlo Simulation and Its Application

Kshetrimayum Newton Singh
Department of Physics & Astrophysics

Contents

- **Introduction**
- **Basic Method and Simple Applications**
  - ❑ ***Computing The Value Of Pi (π)***
  - ❑ ***Monte Carlo Integration***
  - ❑ ***2-D Ising Model***
- **Conclusions**

Introduction

What is Meant by Monte Carlo Method?

- **The term "Monte Carlo method" is used to embrace a wide range of problem solving techniques which use random numbers and the statistics of probability.**
- **The term was coined after the casino in the principality of Monte Carlo. Every game in a casino is a game of chance relying on random events such as ball falling into a particular slot on a roulette wheel, being dealt useful cards from a randomly shuffled deck, or the dice falling the right way!**
- **In principle, any method that uses random numbers to examine some problem is a Monte Carlo method.**

- Monte Carlo methods are nowadays widely used, from the integration of multi-dimensional integrals to solving ab-initio problems in chemistry, physics, medicine, biology etc.

The main requirements of performing Monte Carlo simulation are as follows:-

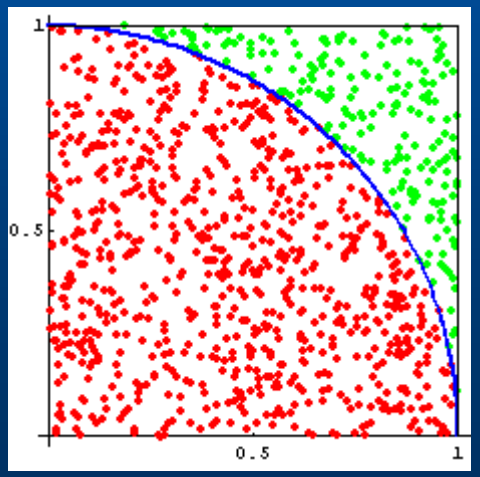
- Be able to generate random variables following a given probability distribution function
- Find a probability distribution function
- Sampling rule for accepting a move
- Compute standard deviation and other expectation values
- Techniques for improving errors

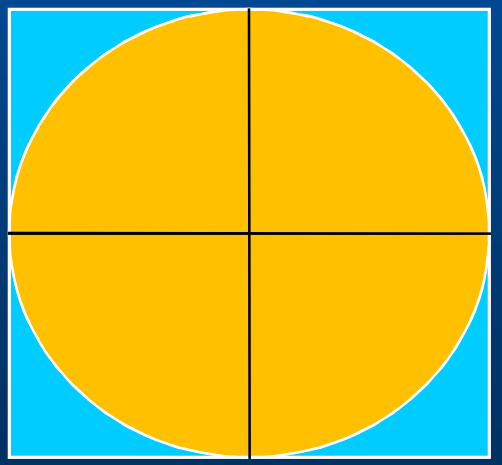
Basic Methods and Simple Application

1. Computing The Value Of Pi (π):

The value of pi can be obtain by finding the area of a circle. In order to find the area of the circle consider the following steps.

- Consider a unit circle which is an in-circle of a square
- Divide the square in 4 quadrants
- Choose the 1st quadrant





- Generate uniform random numbers lies between 0 & 1
- Count the number of the random numbers which fall within the quarter of the circle and the unit square

Since larger the area of those region, the no. of points fall inside it will be larger. Hence we can write

Area of unit Square

$$R^2 \propto N_{sqaure}$$

Area of the quarter of the circle

$$\frac{1}{4}\pi R^2 \propto N_{circle}$$

Now dividing the above equations we get the following equation

$$\pi \approx \frac{4 \times N_{cirlce}}{N_{square}}$$

Now we can find the N_{sqaure} and N_{circle}. To count those number we have the following code:

```
#include<iostream>
#include<ctime>
#include<cmath>
#include<iomanip>
#include<cstdlib>


using namespace std;


  main(void)
  {
  double x_value, y_value;
  long int hit,n_darts=1;

  //generating the seed value for the random numbers
  srand(time(0));

  while(n_darts!=0){

    cout<<"\nThis program estimates Pi by the Monte Carlo method.\n";
    cout<<"How many darts do you want to throw? (0 to quit) ";
    cin>>n_darts;



    if (n_darts!=0)
      {
    //reset the hit variable
    hit=0;
    for(int i=0;i<n_darts;i++)
      {
        //generates a random value of x and y between 0
        //and 1 with the precision of a double variable
        x_value = static_cast<double>(rand())/RAND_MAX;
```

```
        y_value = static_cast<double>(rand())/RAND_MAX;

        //if the x and y values are inside a circlue of
        //unit 1 then    we increase the hit counter
        if (((x_value*x_value)+(y_value*y_value))<=1) hit++;

      }
    //using the setprecision and setioflags functions to
    //set the precision to 10 digits and to show
    //trailing zeros we estimated the value of pi
    //by multiplying the number of hits by 4 and dividing
    //by the numbers of thrown darts.
    cout<<"There were "<<hit<<" hits in the circle \n";
    cout<<"The estimated value of pi is: "
        <<setiosflags(ios::fixed|ios::showpoint)
        <<setprecision(10)
        <<(hit*4)/static_cast<double>(n_darts)<<"\n";

    cout<<"the original value of pi is:"
        <<22.0/7.0;

      }//if (darts!=0)

    }//end of while(n_darts!=0)

}//end of main()
```

```
This program estimates Pi by the Monte Carlo method.
How many darts do you want to throw? (0 to quit) 10000
There were 7857 hits in the circle
The estimated value of pi is: 3.1428000000
the original value of pi is:3.1428571429
This program estimates Pi by the Monte Carlo method.
```

2. Monte Carlo Integration:

The simplest Monte Carlo estimator is very similar to the rectangular quadrature rule setting. Consider the integration

$$I = \int_a^b f(x)\,dx$$

- **Generate uniform random numbers x_i which lies within the interval *[a,b].***
- **Define a random function F_N given by the equation below**

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

Since F_N is a random variable of the integral, we want to find its expected value, and hopefully it could approximate the real value *I* we want:

$$\langle F_N \rangle = \left\langle \left[\frac{b-a}{N} \sum_{i=1}^{N} f(x_i) \right] \right\rangle = \frac{b-a}{N} \sum_{i=1}^{N} \langle f(x_i) \rangle$$

$$= \frac{b-a}{N} \sum_{i=1}^{N} \int_{-\infty}^{+\infty} f(x)\, p(x)\, dx = \frac{b-a}{N} \frac{1}{b-a} \sum_{i=1}^{N} \int_{a}^{b} f(x)\, dx$$

$$= \frac{1}{N} . N \int_{a}^{b} f(x)\, dx = I \qquad p(x) = \frac{1}{b-a} \text{ for } a < x < b$$

Instead of using the uniform sampling, we can use other sampling also. I order to perform this we have the following steps.

Define a new random function F'_N as

$$F'_N = \frac{b-a}{N} \sum_{i=1}^{N} f(x_i) = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{1/(b-a)}$$

$$= \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)}$$

where the denominator is absorbed in $p(x_i)$

Now finding the expectation value of the function

$$\langle F'_N \rangle = \left\langle \left[\frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)} \right] \right\rangle = \frac{1}{N} \sum_{i=1}^{N} \left\langle \frac{f(x_i)}{p(x_i)} \right\rangle$$

$$= \frac{1}{N} \sum_{i=1}^{N} \int_{-\infty}^{+\infty} \frac{f(x)}{p(x)} p(x)\,dx = \frac{1}{N} \sum_{i=1}^{N} \int_{a}^{b} f(x)\,dx \;, a < x < b$$

$$= \frac{1}{N} . N \int_{a}^{b} f(x)\,dx = I$$

Hence now we can see very easily that any type of sampling can be use, but have to be careful about what we have chosen, for a better result.

- **Calculate the error σ given as**

$$\sigma \approx (b-a) \sqrt{\frac{\langle F^2 \rangle - \langle F \rangle^2}{N}}$$

- **Now the final value of the integration is given as**

$$I_{final} = I \pm \sigma$$

A code for calculating the integration is as follows:

```
program mc inte
  implicit none
  integer, parameter :: n=10000000

  call random_seed()  ! needed to initialize the random number generator use
d in MC eval
  call MC_integration(n,3.0)

  contains

    subroutine MC integration(n,end_val)
      implicit none
      integer :: n
      real :: end val
      real :: x, integral, integral_err
      real (kind=8) :: f, f2
      integer :: i

      integral = 0.0

      f= 0.0
      f2 = 0.0

      do i=1,n
         call random number(x)
         x=x*end_val  ! random_number only returns uniformly distributed fro
m [0.0, 1.0]
```

```
        f = f+integrand(x)
        f2 = f2+ (integrand(x)**2)
      end do

      f=f/n
      f2=f2/n

      integral=(end_val-0.0d0)*f
      integral_err= (end_val-0.0d0)*SQRT((f2 - f**2.0d0)/n)

      write (*,*) "# MC integration = ",integral,"+/-",integral_err

    end subroutine MC_integration

    function integrand(x) result (value)
      implicit none
      real :: x
      real :: value

      if (x .lt. 0.00001) then
         x = 0.00001
      end if

      value = exp(x)
    end function integrand

 end program mc_inte
```

```
 # MC integration =    19.082798     +/-  4.89180535E-03


------------------
(program exited with code: 0)
Press return to continue
```

$e = 2.71828$

Reminder from Statistical Mechanics

The *Partition function* contains all thermodynamic information:

$$Z = \sum_{all\ states} e^{-H/k_B T}$$

The probability of the n^{th} state appearing is:

$$P_n = \frac{1}{Z} e^{-H/k_B T}$$

Thermodynamic properties are then determined from the free energy F where

$$F = -k_B T \ln Z$$

3. 2-D Ising model:

Ising model is a statistical model, which was originally used to study the behavior of magnetic particles in a magnetic field. The model consists of a collection of spins on lattice sites. Each spin takes only the value of ***+1*** (up) or ***-1*** (down).

The model has also been used to study alloys, where instead of representing the lattice site as spins it represents what type of atom is in that site. Say if a site is of the A-type atom, than we can assign 1 or otherwise 0, also same condition for B-type atom.

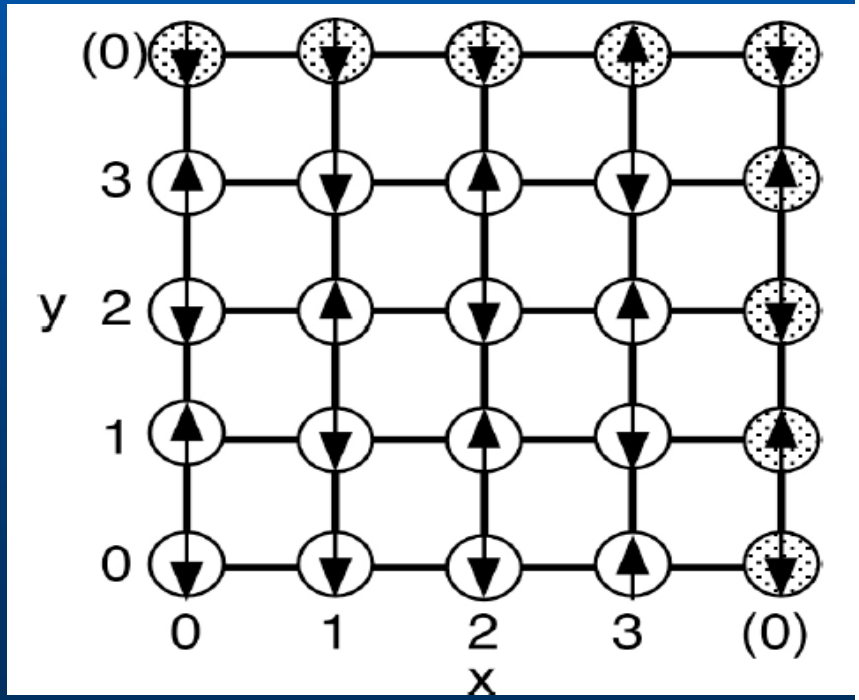
Define a *Hamiltonian* for a system that is dependent on the arrangement of spins on a lattice as

$$H = -J\sum_{<i,j>}\sigma_i\sigma_j - B\mu\sum_i \sigma_i, \quad \sigma_i = \pm 1$$

Each indices consist of a pair of integers, for $L \times L$ lattice

$$i = (\alpha, \beta), \left(0 \le \alpha, \beta \le L-1\right)$$

For a system of N spins have 2^N states!



In order to neglect the surface effect, we consider the periodic boundary condition (pbc).

$$(L,\beta)=(0,\beta) \quad \text{and} \quad (\alpha,L)=(\alpha,0)$$

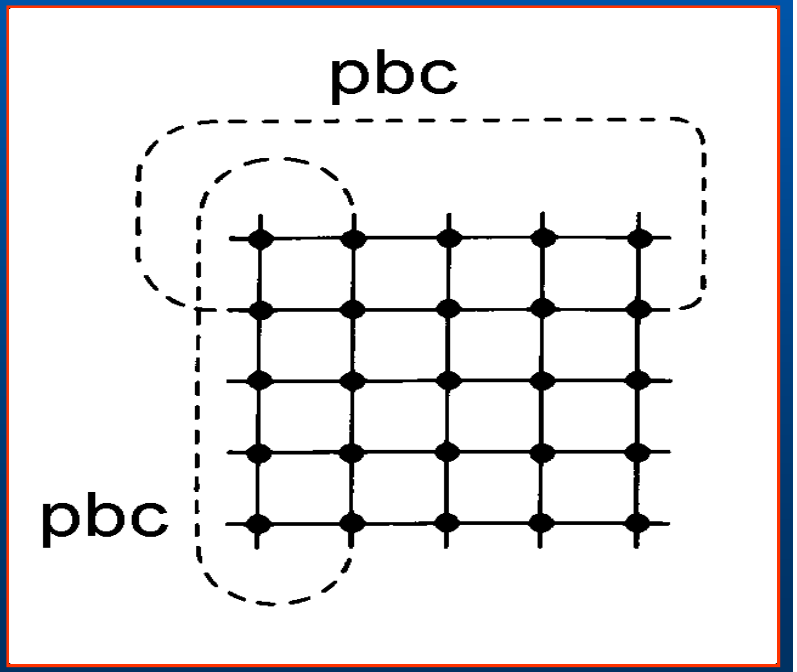
which means that the grid points (L-1,β) & (0,β) are east-west neighbors and (α,L-1) & (α,0) are north-south neighbors.

Theoretically, such a system can be solved by first solving the partition function given as

$$Z=\sum_{all\ states} e^{-H/k_BT}$$

where

$$H=-J\sum_{<i,j>}\sigma_i\sigma_j-B\mu\sum_i\sigma_i, \quad \sigma_i=\pm 1$$



Then the probability of finding the system in a certain state, denoted n^{th} is

$$P_n = \frac{1}{Z} e^{-H / k_B T}$$

Now we can find the other physical quantities

Magnetization, $$M = \frac{\partial \ln Z}{\partial B} = \sum_n P_n \sum_i S_i$$

Susceptibility, $$\chi = \frac{\partial M}{\partial B} = \sum_n P_n \left(\sum_i S_i \right)^2 - M^2$$

Internal Energy, $$E = \sum_n p_n H_n$$

Specific Heat, $$C = \sum_n p_n H_n^{\ 2} - E^2$$

For a zero external magnetic field, *B=0* we have

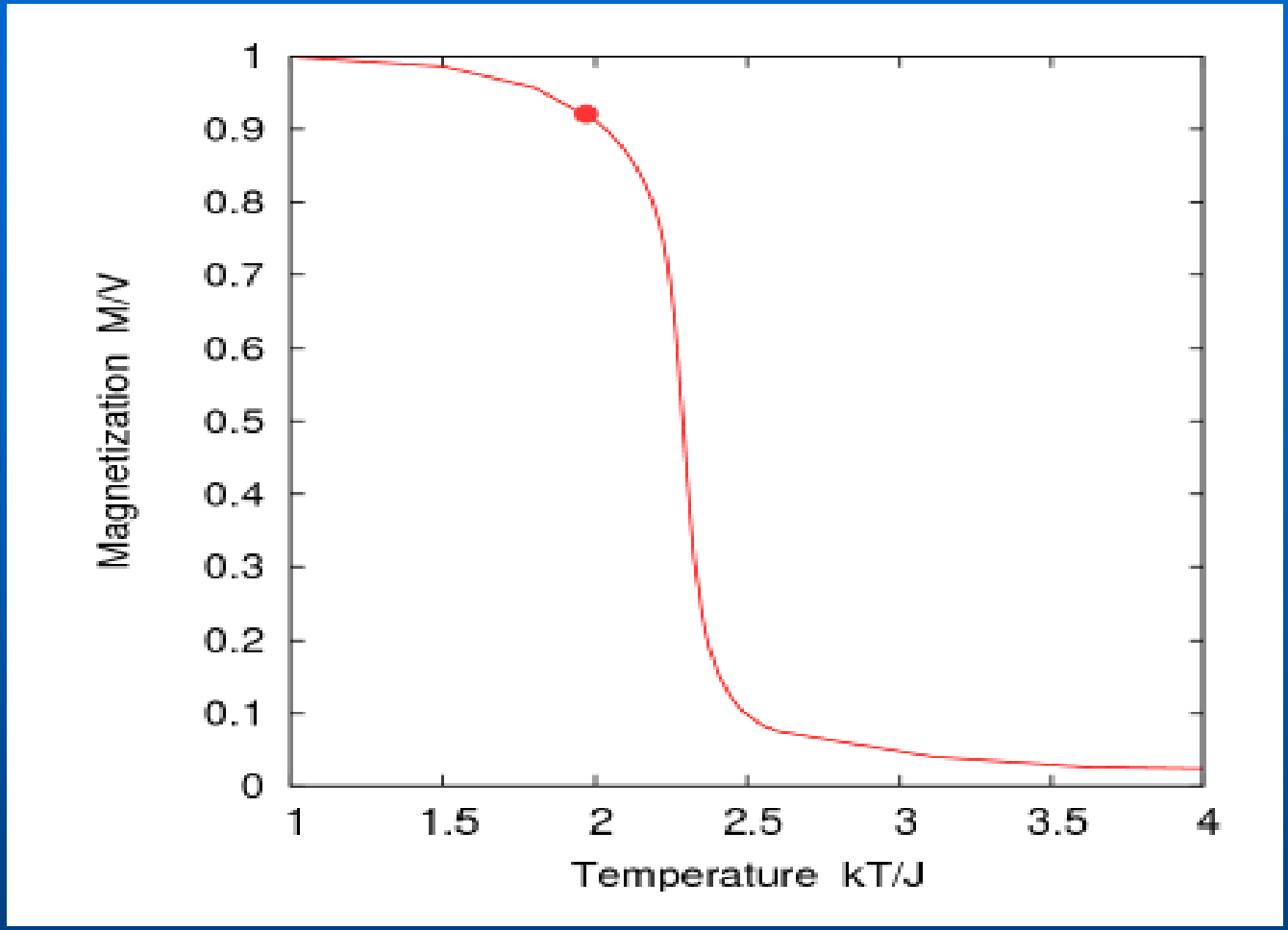
$$H = -J \sum_{<i,j>} \sigma_i \sigma_j, \quad \sigma_i = \pm 1$$

Even if the external field is not applied, there is still a net magnetization if we allow it to decrease the temperature to a certain value. The value of T_c is given below

$$\frac{k_B T_c}{J} = \frac{2}{\ln\{1+\sqrt{2}\}} \approx 2.269$$

This spontaneous magnetization happens after at a certain temperature is due to the occurrence of phase transition from paramagnet to ferromagnet.

$$M_0 \approx \sqrt{3\left(1-\frac{T}{T_c}\right)}, 0 < 1-\frac{T}{T_c} << 1$$



This figure shows that there is non-zero magnetization below the critical temperature T_c

In order to solve the above problem, we can use the Monte Carlo Simulation.

Single spin-flip sampling for the Ising model

Produce the n^{th} state from the m^{th} state … relative probability is $P_n/P_m \rightarrow$ need only the *energy difference, i.e.* $\Delta E = (E_n - E_m)$ between the states

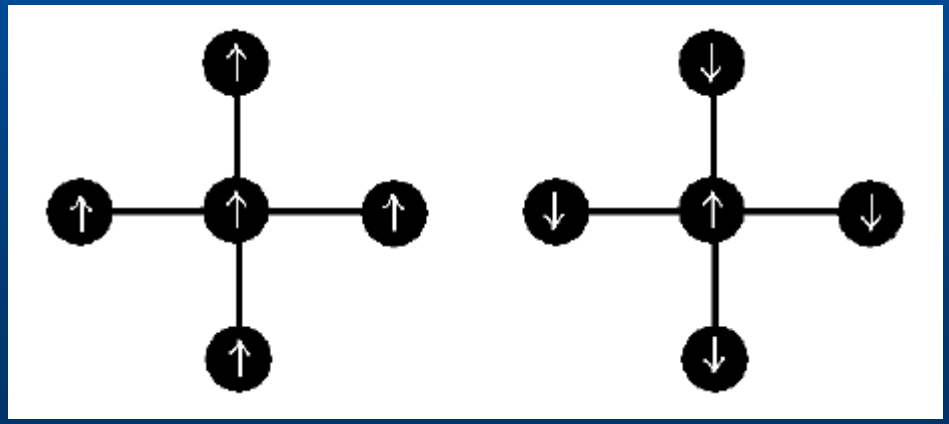
Any transition rate that satisfies *detailed balance* is acceptable, usually the Metropolis form

$$W(m \rightarrow n) = \tau_o^{-1} \exp(-\Delta E/k_B T), \quad \Delta E > 0$$
$$= \tau_o^{-1}, \quad \Delta E < 0$$

where τ_o is the time required to attempt a spin-flip.

Metropolis Recipe:

1. Choose an initial state
2. Choose a site *i*
3. Calculate the energy change ΔE that results if the spin at site *i* is overturned
4. Generate a random number *r* such that $0 < r < 1$
5. If $r < \exp(-\Delta E/k_B T)$, flip the spin
6. Go to 2



The code of solving the 2-D Ising Model

```
program ferromagnetism
integer lbox,ntime,itime,phi,i,j,nsite,isite,idum,ntemp,itemp
real r,w,T,ran2,a,b,de,u,v
parameter (lbox = 256)
dimension phi(lbox,lbox),phin(lbox,lbox),rr(lbox),ll(lbox)

nsite = lbox**2
ntime = 5000
idum = -1234567
ntemp = 40
T = 0.0

!Setting of Periodic Boundary Conditions
do i = 1 , lbox
  rr(i) = i + 1
  ll(i) = i - 1
end do
rr(lbox) = 1
ll(1) = lbox

!Setting of the initial values of the order parameter
do i = 1 , lbox
  do j = 1 , lbox
     r = 2.0*ran2(idum) - 1.0
     if(r.gt.0.0)then
     phin(i,j) = 1
     else
     phin(i,j) = -1
     end if
  end do
end do
```

```
!Variation of Temperatute's
do itemp = 1 , ntemp
T = T + 0.1

  do i = 1 , lbox
    do j = 1 , lbox
      phi(i,j) = phin(i,j)
    end do
  end do

  do itime = 1 , ntime
    do isite = 1 , nsite

i = 1 + (lbox - 1)*ran2(idum)
j = 1 + (lbox - 1)*ran2(idum)
a = -phi(i,j)*(phi(rr(i),j)+phi(ll(i),j)+phi(i,rr(j))+phi(i,ll(j)))
b = +phi(i,j)*(phi(rr(i),j)+phi(ll(i),j)+phi(i,rr(j))+phi(i,ll(j)))
      de = b - a

      if(de.le.0.0)then
        phi(i,j) = - phi(i,j)
      else
        w = exp(-de/T)
        r = ran2(idum)
        if(w.gt.r)then
          phi(i,j) = - phi(i,j)
        else
        end if
      end if
```

```
    end do
  end do

  u = 0.0
  do i = 1 , lbox
    do j = 1 , lbox
      v = real(phi(i,j))
      u = u + (v/nsite)
    end do
  end do

  write(10,*)T,abs(u)

end do

end program ferromagnetism
```

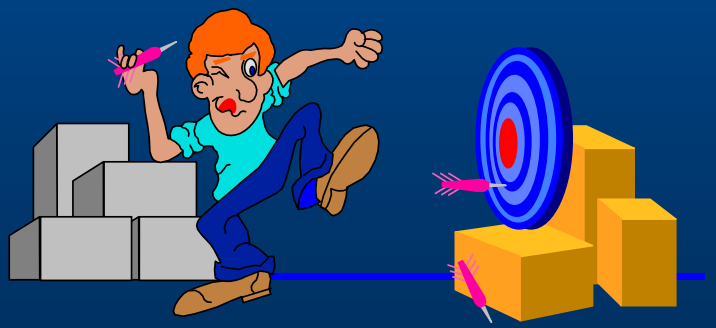
```
FUNCTION ran2(idum)
INTEGER idum,IM1,IM2,IMM1,IA1,IA2,IQ1,IQ2,IR1,IR2,NTAB,NDIV
REAL ran2,AM,EPS,RNMX
PARAMETER (IM1=2147483563,IM2=2147483399,AM=1./IM1,IMM1=IM1-1,IA1=40014)
PARAMETER (IA2=40692,IQ1=53668,IQ2=52774,IR1=12211,IR2=3791,NTAB=32)
PARAMETER (NDIV=1+IMM1/NTAB,EPS=1.2e-7,RNMX=1.-EPS)
INTEGER idum2,j,k,iv(NTAB),iy
SAVE iv,iy,idum2
DATA idum2/123456789/, iv/NTAB*0/, iy/0/
if (idum.le.0) then
idum=max(-idum,1)
idum2=idum
  do 11 j=NTAB+8,1,-1
     k=idum/IQ1
     idum=IA1*(idum-k*IQ1)-k*IR1
     if (idum.lt.0) idum=idum+IM1
     if (j.le.NTAB) iv(j)=idum
```

```
11     continue
       iy=iv(1)
       endif
k=idum/IQ1
idum=IA1*(idum-k*IQ1)-k*IR1
if (idum.lt.0) idum=idum+IM1
k=idum2/IQ2
idum2=IA2*(idum2-k*IQ2)-k*IR2
if (idum2.lt.0) idum2=idum2+IM2
j=1+iy/NDIV
iy=iv(j)-idum2
iv(j)=idum
if(iy.lt.1)iy=iy+IMM1
ran2=min(AM*iy,RNMX)
return
END
```

Types of Computer Simulations

Deterministic methods . . . (Molecular dynamics)

Stochastic methods . . . (*Monte Carlo*)



Conclusions

- Makes it possible to study more complicated models, which do not have an analytical solution (or solution is difficult).
- Don’t have to make as many simplifying assumptions- get more flexible models that can be more valid.
- Can include randomness in a controlled way
- Don’t get exact answers-only estimates, which Include uncertainty - that should also be estimated