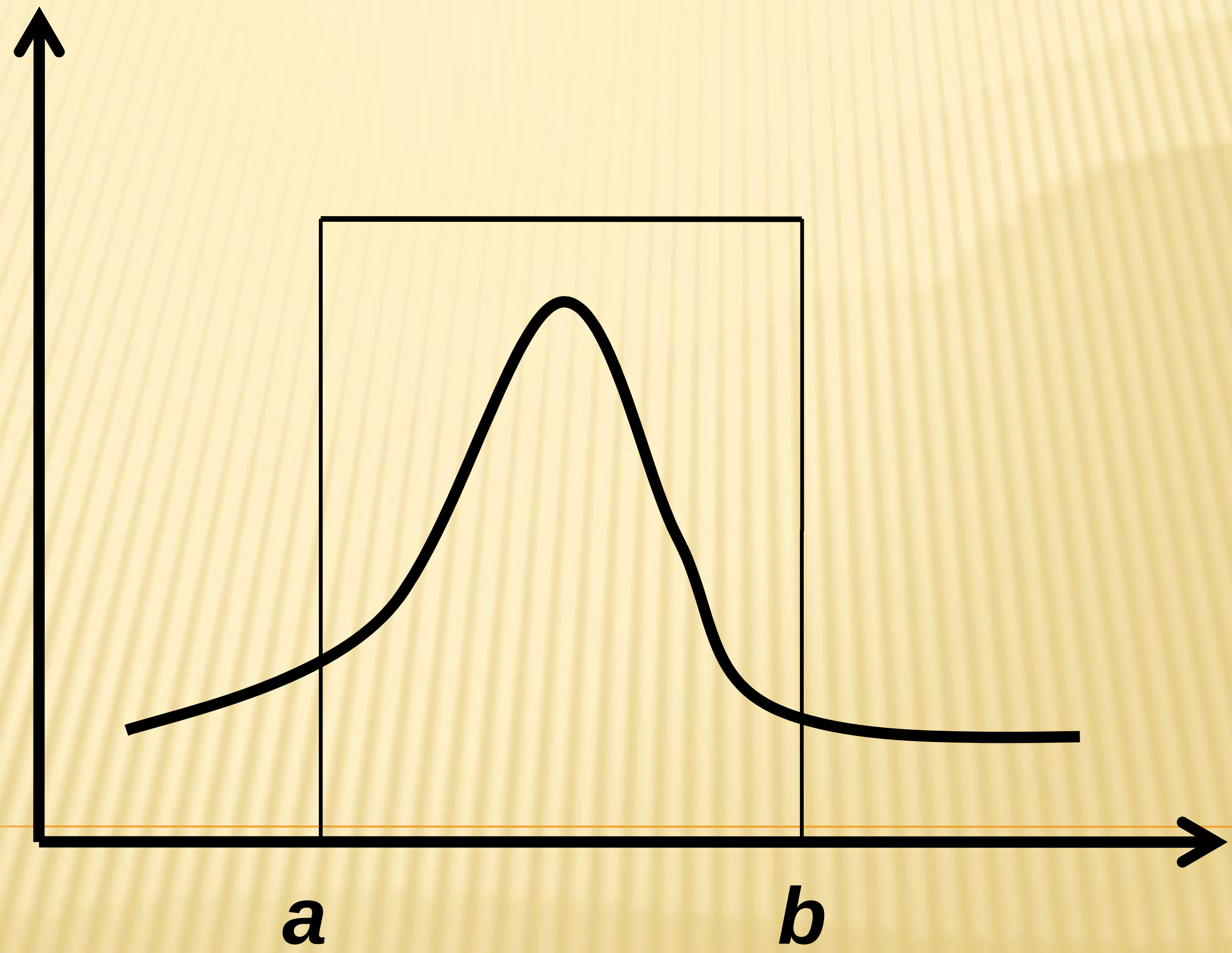
2010.5

计算物理·物理模拟II

COMPUTATIONAL PHYSICS·COMPUTER SIMULATION

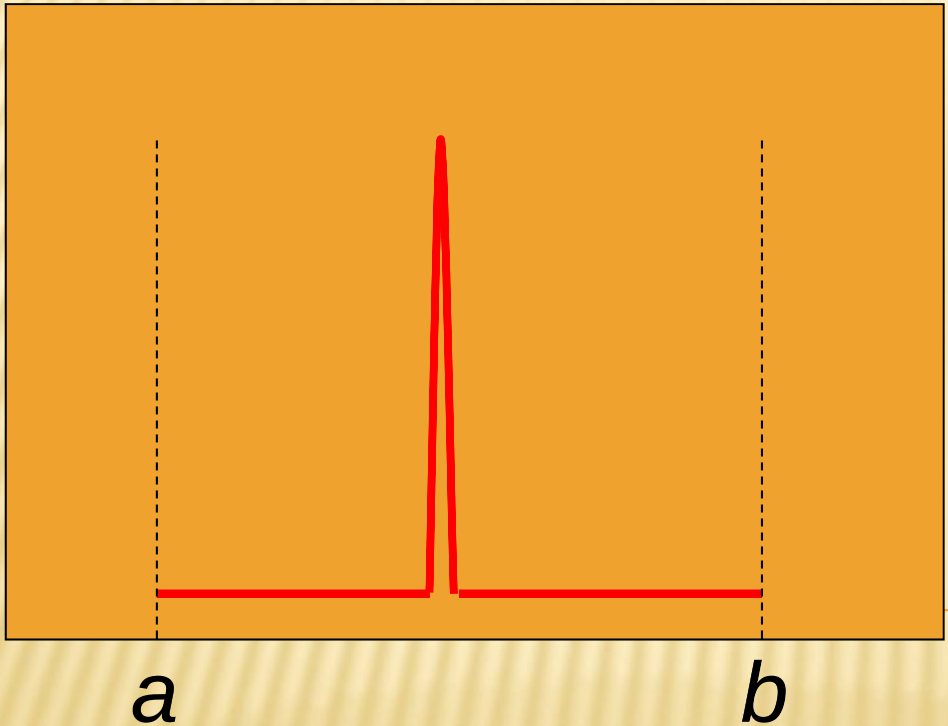
Monte Carlo方法计算积分：



如果，我们遇到一个 变态。。。

。。。的函数

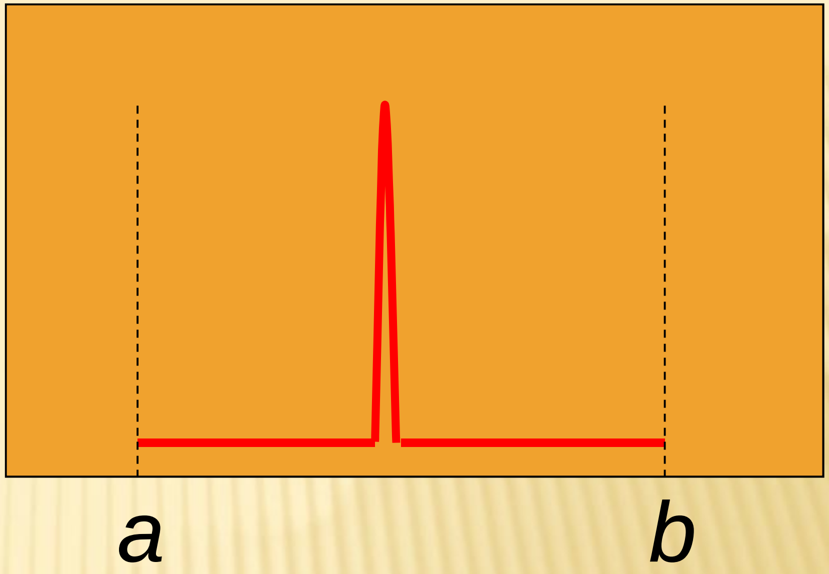
问题，求如下函数曲线下的面积。



如果用一个均匀分布的随机数，会出现什么情况？

实际问题中，往往比这个函数更变态！

Solution



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

$$\Rightarrow I = \int_a^b \frac{f(x)}{w(x)} w(x)dx$$

使重要的部分更为突出 → **Importance Sampling**

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

$$where \quad f^*(x) = \frac{f(x)}{w(x)}$$

w(x)也被称为权重因子。

如 **w(x)**是归一化的，即 $\boxed{\int_a^b w(x)dx = 1}$

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

w(x)dx为概率，***I*** 为 ***f**(x)**的平均值。

如按照分布**w(x)**生成随机数 ξ

$$w(x)dx = n/N$$

其中**N**为随机数总个数，**n**为落在**w(x)**区间的随机数的个数。

$$\boxed{I = E\{f^*(\xi)\} \approx \frac{1}{N}\sum_{i=1}^{N} f^*(\xi_i)}$$

ξ_i 是以**w(x)**分布的随机数

具体步骤:

1）确定重要性采样的分布**w(x)**，并归一化。

$$\int_a^b w(x)dx = 1$$

2）按照分布**w(x)**生成随机数 ξ，计算

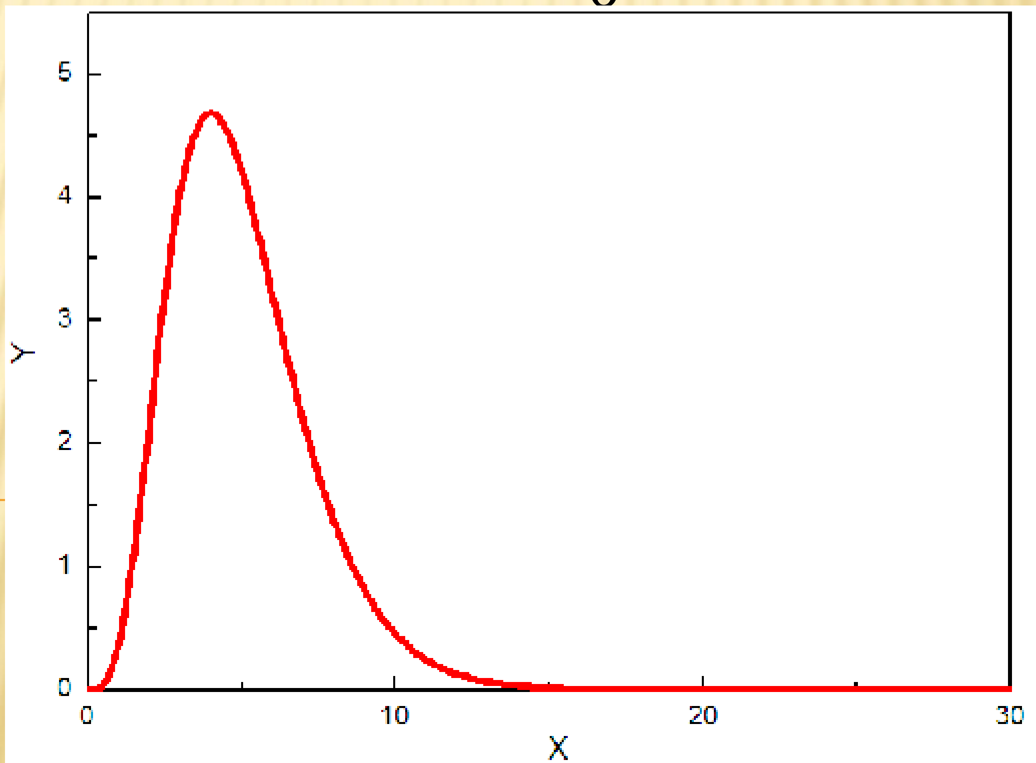
$$I = \int_a^b \frac{f(x)}{w(x)} w(x)dx = \frac{1}{N}\sum_{i=1}^{N} \frac{f(\xi_i)}{w(\xi_i)}$$

其中N为随机数总个数。

例题：

用重要性抽样法来计算积分：

$$I = \int_0^{\infty} x^4 e^{-x} dx = 4! \int_0^{\infty} e^{-x} dx = 24$$



候选权重函数

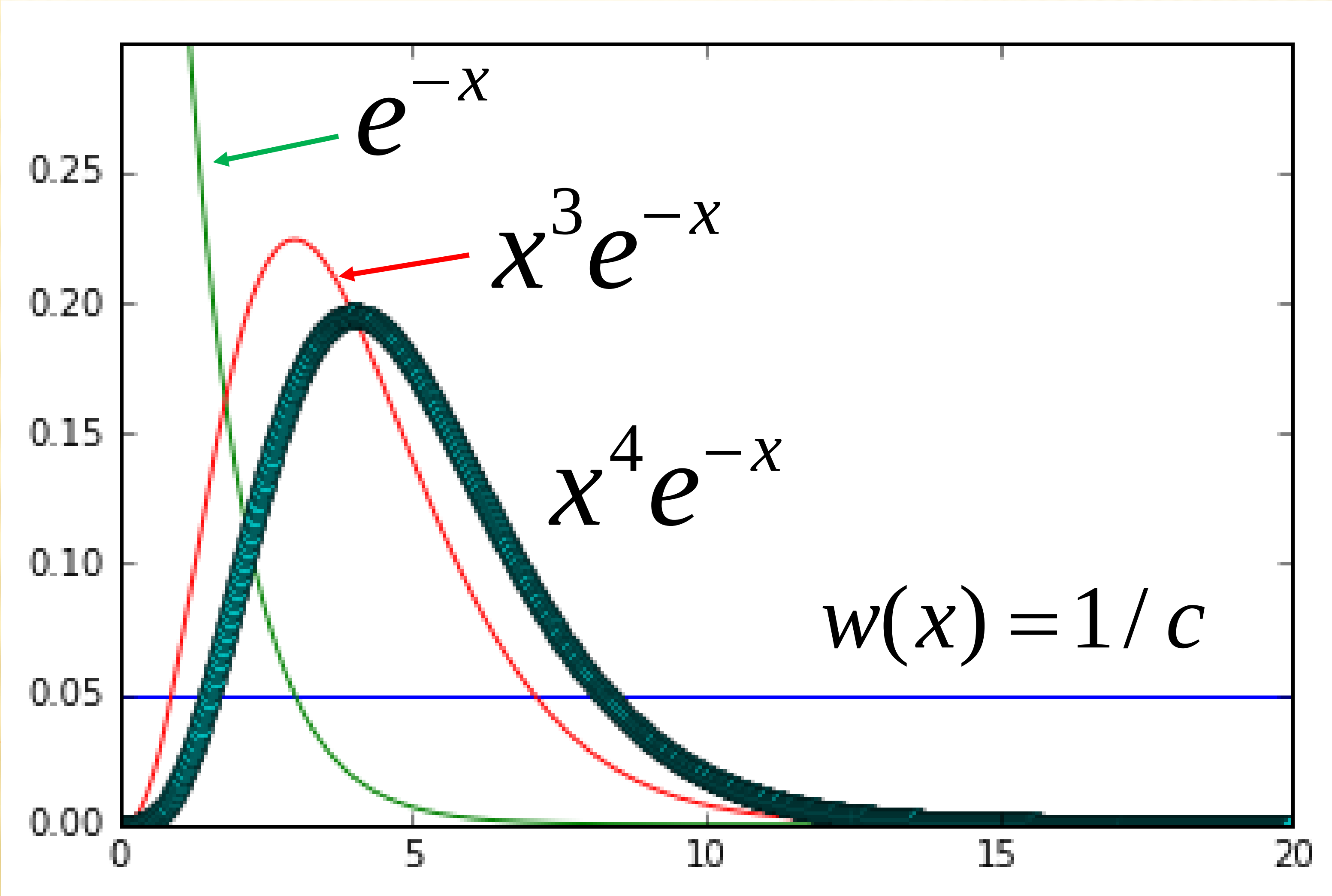
1) $$w(x)=1/c$$

2) $$w(x)=e^{-x}$$

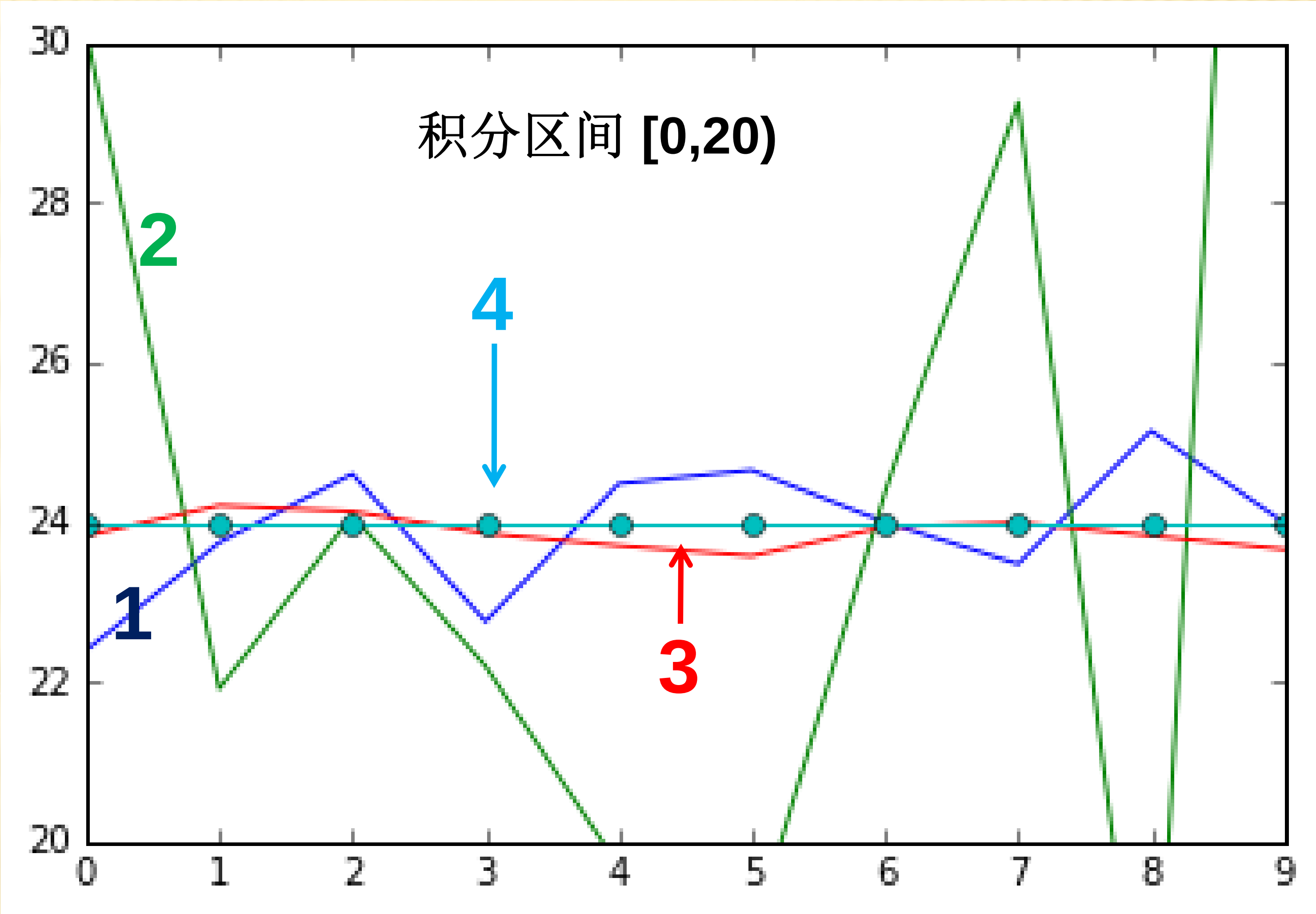
3) $$w(x)=\frac{1}{6}x^3e^{-x}$$

4) $$w(x)=\frac{1}{24}x^4e^{-x}$$

候选权重函数



Result 1: 22.39 23.74 24.60 22.75 24.48 24.65 24.00 23.47 25.14 23.99
Result 2: 30.16 21.91 24.08 22.20 19.70 18.76 24.37 29.26 16.32 44.43
Result 3: 23.83 24.21 24.13 23.86 23.70 23.58 23.94 23.99 23.83 23.66
Result 4: 24.00 24.00 24.00 24.00 24.00 24.00 24.00 24.00 24.00 24.00



结论：

1. 重要性抽样好。

2. 抽样分布应该尽量接近被积函数。

课堂练习：

```
AA = 0.0
BB = 20.0

def ff(x):
    x2 = x*x
    return x2*x2*np.exp(-x)

def wt1(x):
    return 1.0/(BB-AA)

def wt2(x):
    return np.exp(-x)

def wt3(x):
    x2 = x*x
    return np.exp(-x)*x2*x/6.0

def wt4(x):
    x2 = x*x
    return np.exp(-x)*x2*x2/24.0
```

```
max1 = 1.0/(BB-AA)
max2 = 1.0
max3 = 0.3
max4 = 0.3

wt = wt4  #change when necessary
wtmax= max4  #change when necessary

nt = 1000
list = np.zeros(nt)
ic = 0
while(ic<nt):
    …
    …
pl.hist(list,bins=20)
pl.show()

II = 0.0
for i in range(len(list)):
    x = list[i]
    II += ff(x)/wt(x)
II /= len(list)
print("the integral result is: ",II)
```

作业：

用重要性抽样法来计算积分：

$$I = \int_0^{20} x^4 e^{-x} dx$$

取： $w(x) = 1/20$, 和 $w(x) = x^3 e^{-x} / 6$

分别对两个权重函数，计算多次计算结果的:

平均值、标准偏差、标准误差。

并作比较。递交程序和相关PDF文件。

规范邮件标题：作业7-姓名-学号

Re-think

$$w(x) = \frac{1}{24} x^4 e^{-x}$$

if $w(x) = \dfrac{1}{24} x^4 e^{-x}$

$$I = \int_0^\infty \frac{f(x)}{w(x)} w(x)dx = 24\int_0^\infty 1 \bullet w(x)dx$$

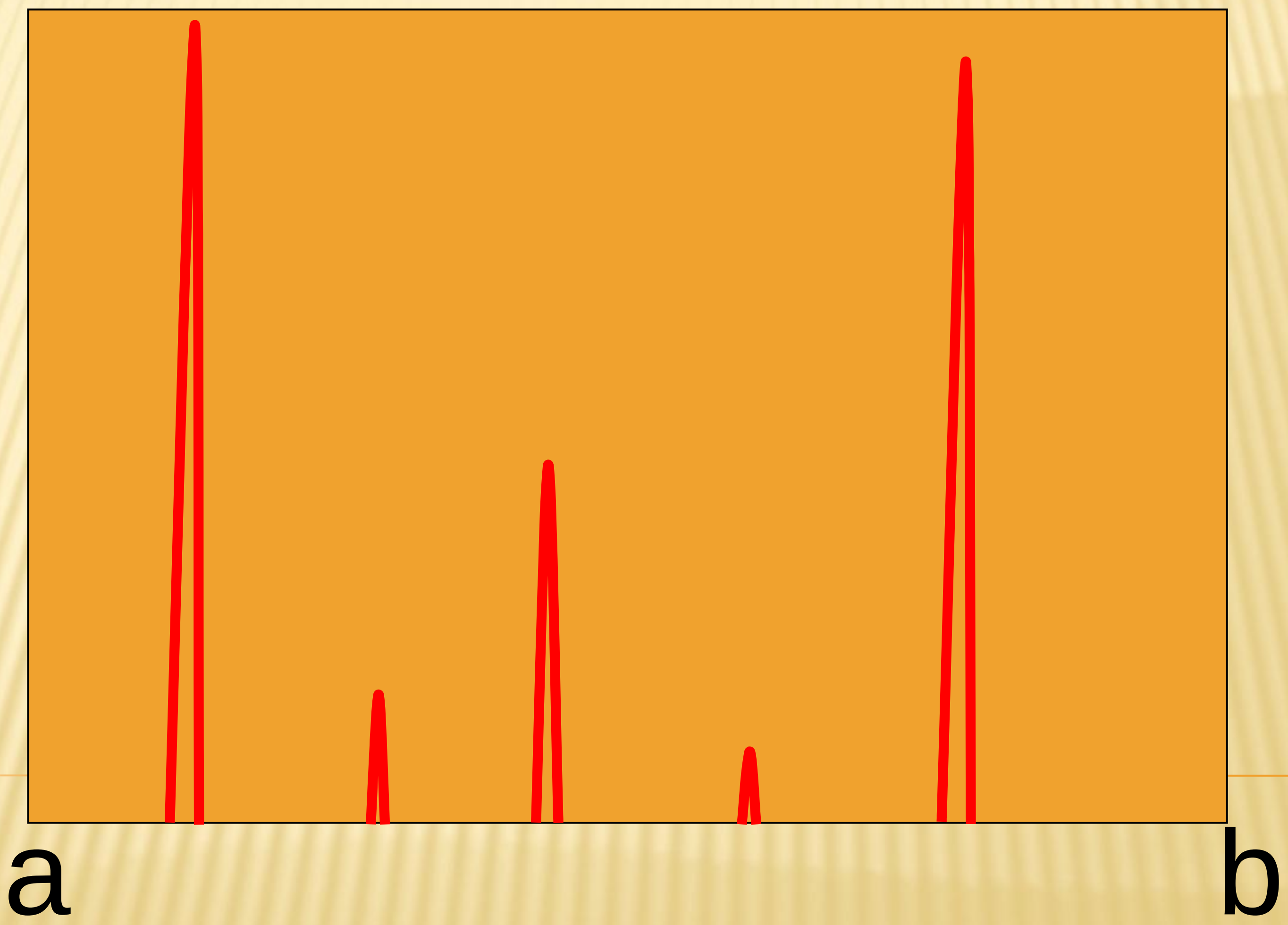
$$= 24 \frac{1}{N} \sum_i 1$$

where the distribution of *i* is *w(x)*

Always gives 24!

赖皮？！

Think about this function

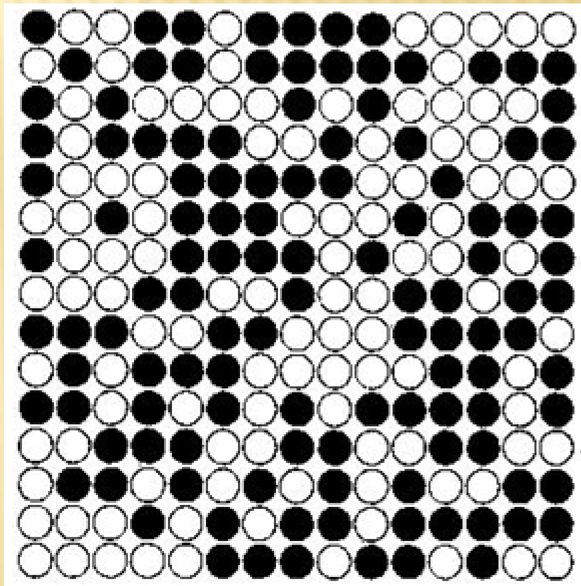


Ising model – a physical case

Spin Interaction

$$H = -\sum_{ij} J_{ij} S_i S_j \qquad H = -\sum_{ij} J_{ij} S_i S_j + \sum_i h_i S_i$$

Boltzmann Distribution



$$P(S) \propto e^{-\beta E(S)}, \beta = 1/k_B T$$

Physical quantities:

average energy,

magnetism, ….

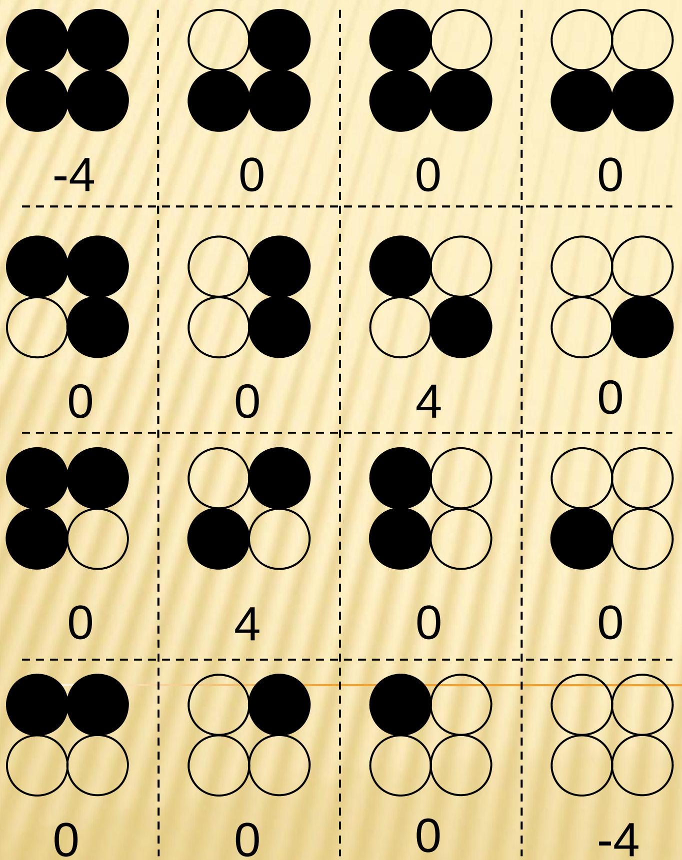
$$< E >= \sum E(S) e^{-\beta E(S)} / Z, \qquad Z = \sum e^{-\beta E(S)}$$

$$M(S) = N_{up}(S) - N_{down}(S)$$

为方便，以后暂时略写归一化因子

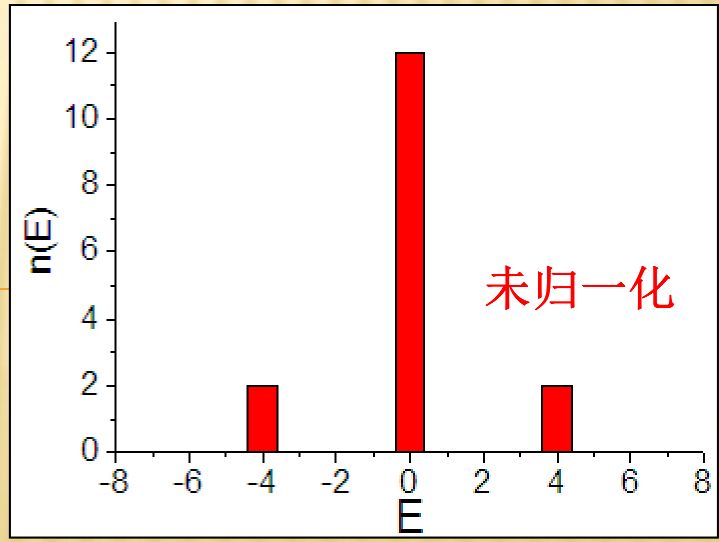
First – Calculate by enumeration

A 2 X 2 case, totally 2^4=16 configurations.

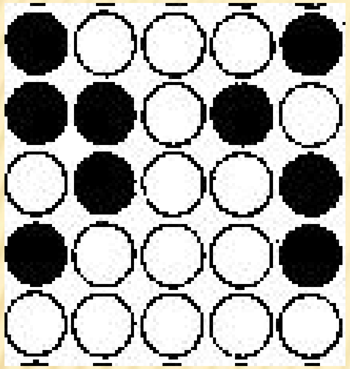


$$H = -\sum_{ij} S_i S_j \qquad S_i = -1, or +1$$

$$< E >= \sum_i E_i e^{-\beta E_i} = \int_0^\infty E \cdot \underline{n(E)} e^{-\beta E} dE$$



Calculate by enumeration (5x5 case)

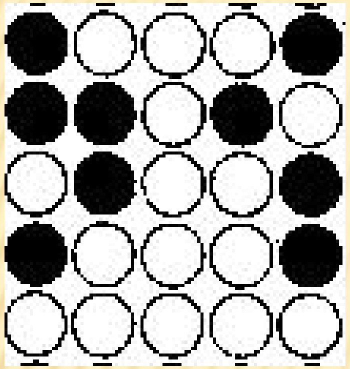


$$E_{groundstate} = -2N(N-1) = -40$$

$$< E > = \sum_i E_i e^{-\beta E_i} = \int_0^\infty E \cdot n(E) e^{-\beta E} dE$$

程序。。。

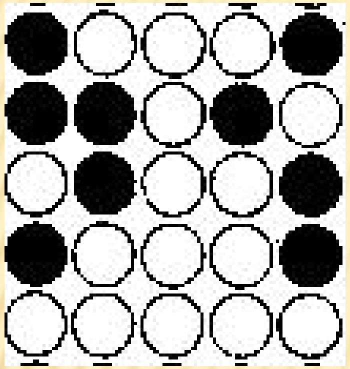
Calculate by enumeration (5x5 case)



```
void enumeration()
{
  histogram hsg;

  int M[N][N];
  for(int i=0;i<N;i++)
  for(int j=0;j<N;j++)
    M[i][j]=-1;

  while(1) {
    int ix=0, iy=0, tag=0;
    while(1) {
      if(M[ix][iy]==-1) {
        M[ix][iy]=1;
        //record this configuration and energy
        break; //find a new configuration
      } else {
        M[ix][iy]=-1;
        iy++;
        if(iy==N) {
          iy=0; ix++;
        }
      }
      if(ix==N) {tag=1;break;}
    }
    if(tag==1) break;
  }
```

运行时间：约**5**秒

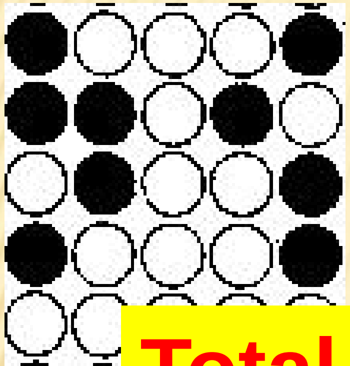
Calculate 5x5 case)



```
import numpy as np
import pylab as pl
import time

N = 5
NN = N*N
M = np.zeros(NN,dtype=int)

I = np.int64(0)
IMAX = np.power(2,NN) - 1
print(bin(I),bin(IMAX))

t1 = time.clock()
while (I<=IMAX) :
   ss = bin(I)
   length = len(ss)-2
   for i in range( length ):
      M[NN-length+i] = ss[i+2]
   I = I + 1
   if(I%50000==0):
      print(M)
pass
t2 = time.clock()
print("running time: ",t2-t1)
```

('running time: ', 656.79)

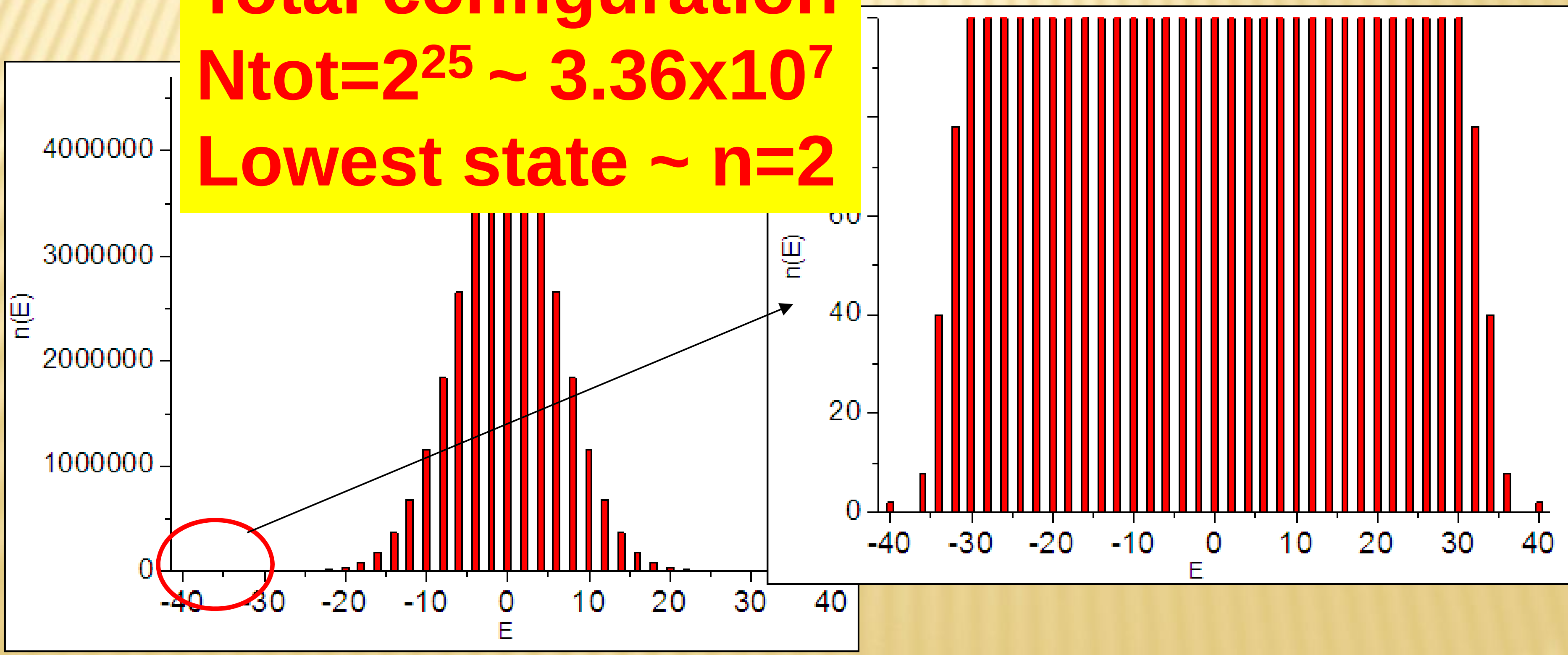
Calculate by enumeration (5x5 case)



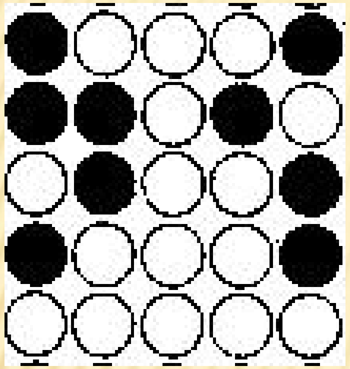
$$E_{groundstate} = -2N(N-1) = -40$$

$$< E >= \sum E_i e^{-\beta E_i} = \int_0^{\infty} E \cdot n(E) e^{-\beta E} dE$$

Total configuration
Ntot=2^{25} ~ 3.36x10^7
Lowest state ~ n=2

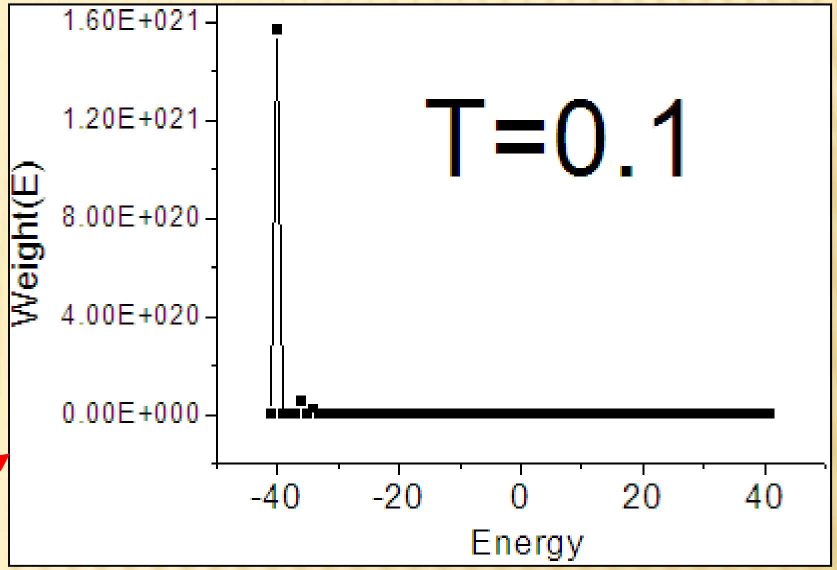


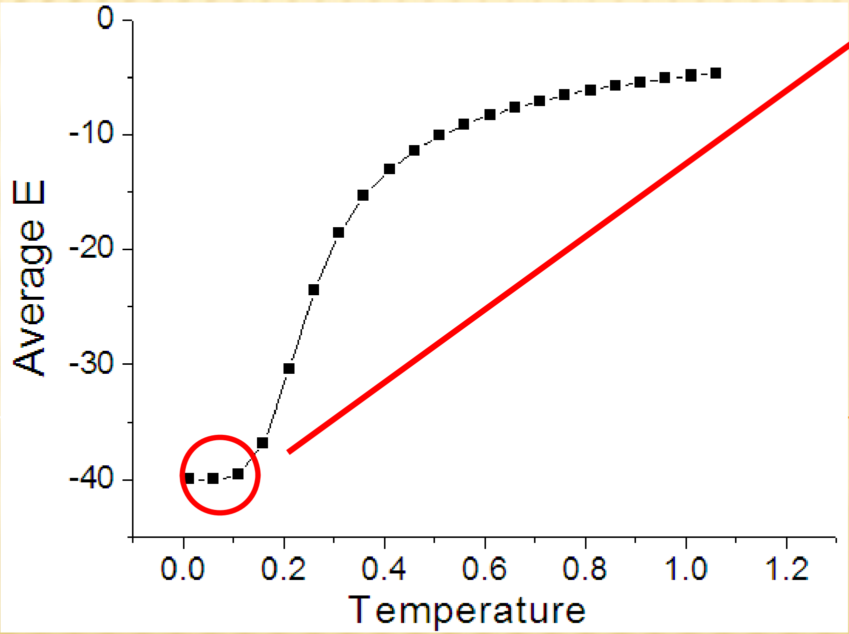
Calculate by enumeration (5x5 case)

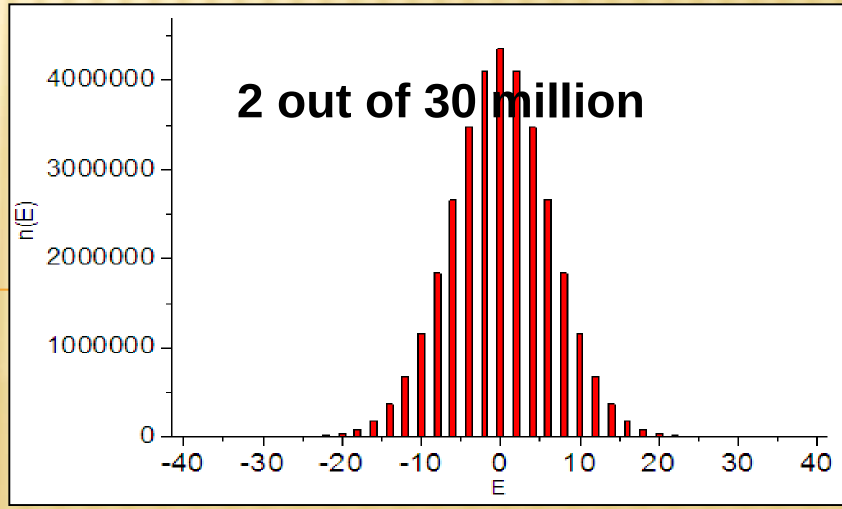


$$< E >= \sum_i E_i e^{-\beta E_i}$$

$$= \int_0^\infty E \cdot n(E) e^{-\beta E} dE$$





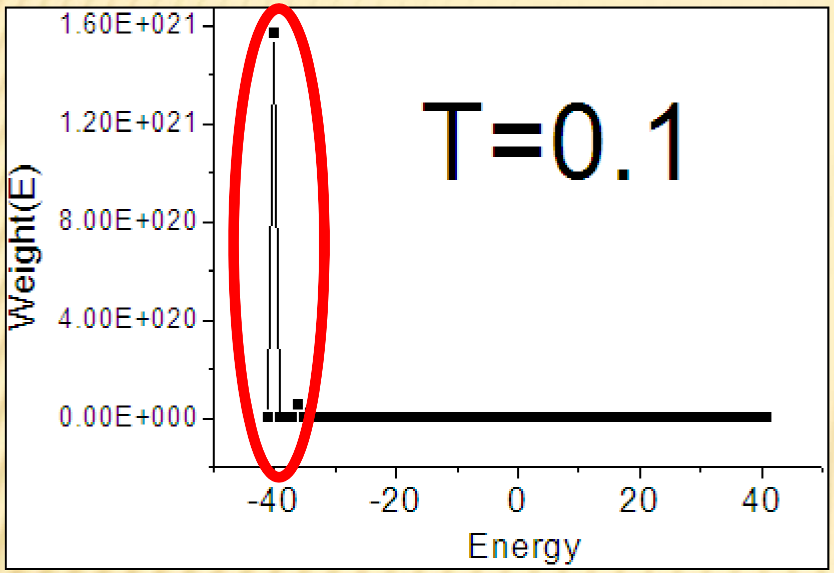


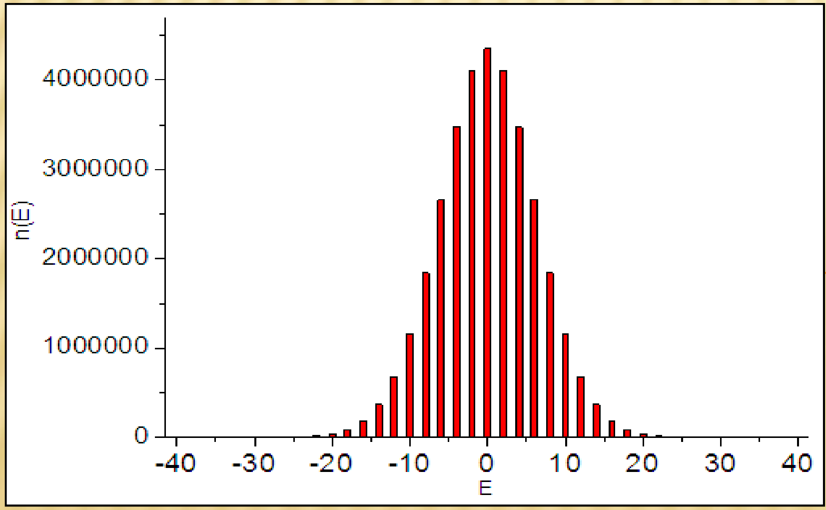
Calculate by enumeration (10x10 case)

Total number of configuration = $2^{100} \sim 10^{30}$

- **If a 5x5 case needs 0.1 second to finish for a modern CPU**
- **The 10x10 case will need $10^{30}/10^{7}$ x 0.1 = 10^{22} seconds**
- **It is about 10^{15} year!**
- **The age of the universe up to now is 10^{10} year**
- **This is only for a 10x10 case!**

Solution – Importance sampling





$$< E >= \int_0^{\infty} E \cdot n(E) e^{-\beta E} dE$$

Instead of using a <u>uniform</u> sampling, we use a <u>biased</u> sampling.

Solution – Importance sampling

$$<E>=\sum_i E_i e^{-\beta E_i}=\sum_i w_i \cdot E_i \frac{e^{-\beta E_i}}{w_i}$$

If we set $w_i = e^{-\beta E_i}$

$$<E>=\sum_i e^{-\beta E_i} \cdot E_i \cdot \frac{e^{-\beta E_i}}{e^{-\beta E_i}}=\sum_i e^{-\beta E_i} \cdot E_i \cdot 1$$

$$<E>=\sum_\zeta E_\zeta / N$$

随机数ξ满足的分布为 $e^{-\beta E_i}$

如何生成分布？

随机数ξ满足的分布为 $e^{-\beta E_i}$

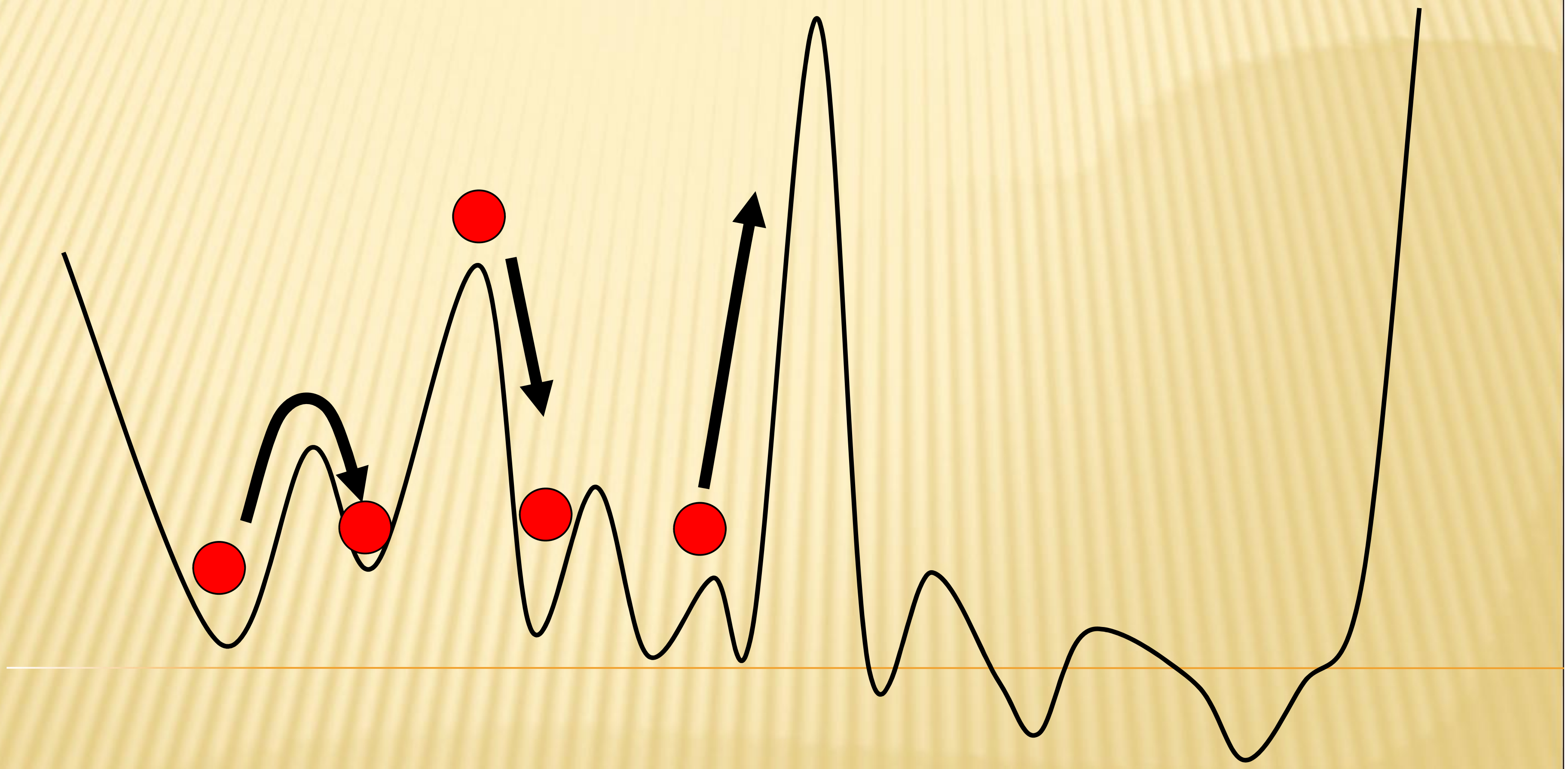
反函数法?
舍选法?
Metropolis procedure

Metropolis Procedure

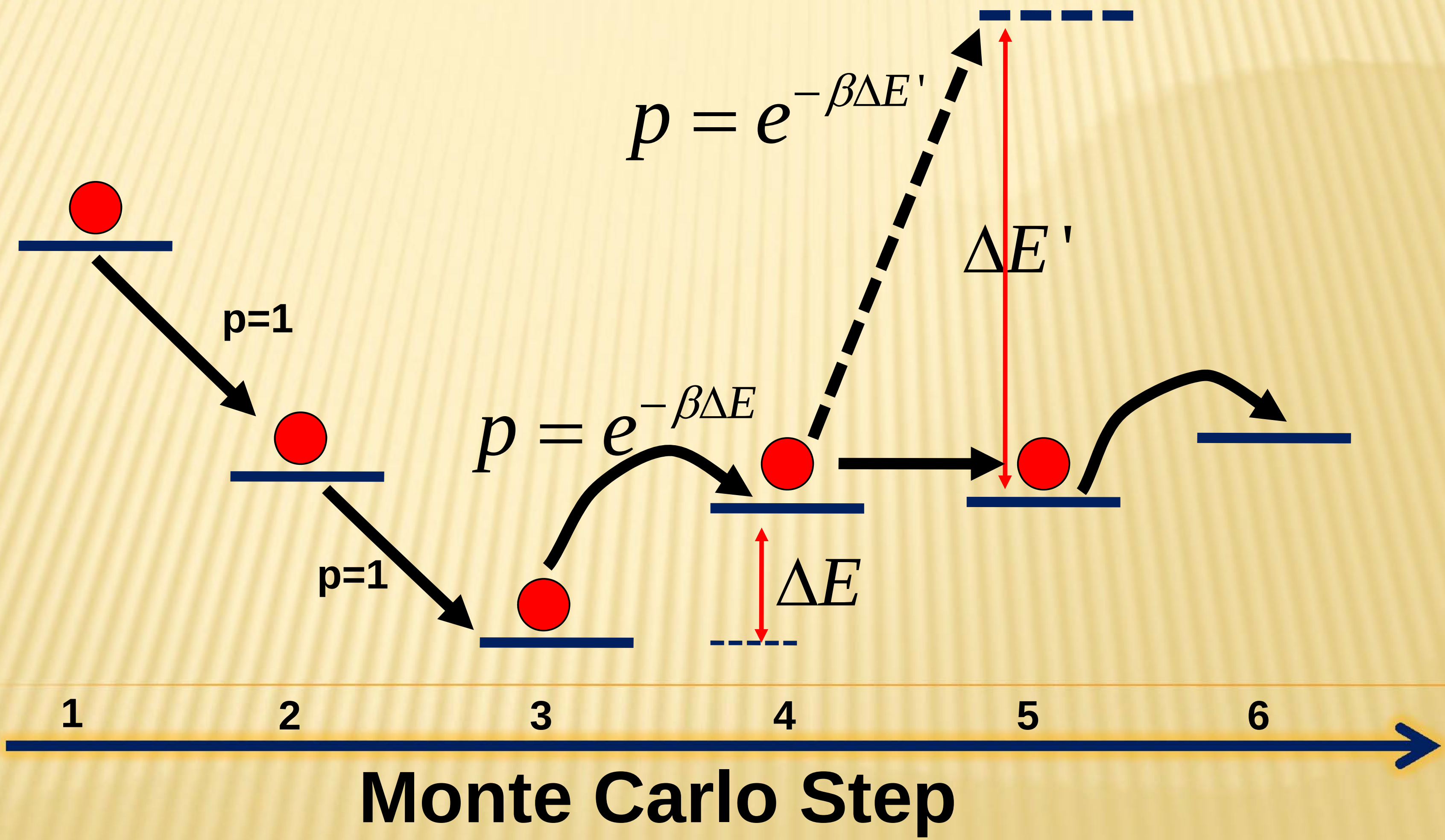
- 1. generate one state i
- 2. calculate its energy → Ei
- 3. generate another state j and calc its Ej
- 4. compare Ei and Ej
- 5. if E decreased, i.e. Ei>Ej, accept and record the state j with the probability 1, and set i=j

  if E increased, i.e. Ei<Ej, accept it with the probability exp(-ΔE/RT), where ΔE=Ej-Ei, and set i=j if successful
- 6. go to 3

Metropolis Monte Carlo



Metropolis Monte Carlo



Why does it work?

1. Consider two states of the system, i and j,
2. Two transition probability w(i→j) and w(j→i), following the Metropolis criterion,
3. Now calculate the equilibrium probability Pi and Pj.

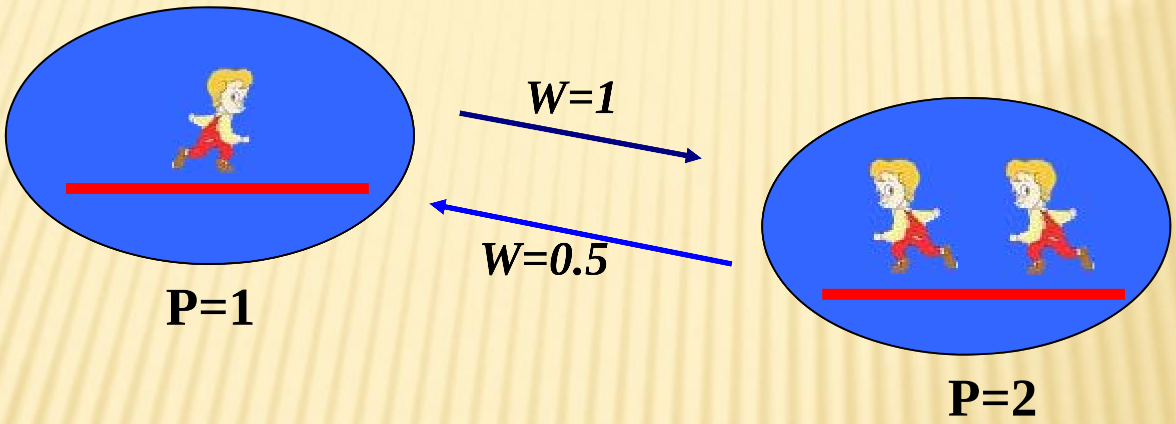
In the equilibrium state, we have

$$P_i w(i \to j) = P_j w(j \to i)$$

$$w(i \to j) = \begin{cases} e^{-\beta(E_j - E_i)} & (E_j > E_i) \\ 1 & (E_j < E_i) \end{cases} = \min(1, e^{-\beta(E_j - E_i)})$$

Detailed Balance (细致平衡)

$$P_i w(i \to j) = P_j w(j \to i)$$



- Detailed balance guarantees a thermal equilibrium state in principle
  - → no change of probability
  - → no net flux between any two states

$$\boxed{P_i w(i \to j) = P_j w(j \to i)}$$

$$w(i \to j) = \begin{cases} e^{-\beta(E_j - E_i)} & (E_j > E_i) \\ 1 & (E_j < E_i) \end{cases} = \min(1, e^{-\beta(E_j - E_i)})$$

If $E_j < E_i \qquad P_i / P_j = e^{-\beta E_i} / e^{-\beta E_j}$

If $E_j > E_i \qquad P_i / P_j = e^{-\beta E_i} / e^{-\beta E_j}$

到达平衡态后，状态分布为 $e^{-\beta E_i}$