| 姓名 | 学号 | 班级 | 选题 | 论述 | 总结 | 得分 |
| --- | --- | --- | --- | --- | --- | --- |
| 李尧 | 2013301020048 | 天眷班 | | | | |

# 标题：Ising 模型的蒙特卡罗模拟

作者：李　　尧

学号：2013301020048

班级：天 眷 班

**摘要**：Ising 模型是最简单的物理模型之一，但是能很好的描绘许多物理过程，例如铁磁相变，合金有序到无序的转变，液氦从常态到超流态的转变等等。遗憾的是，这样一个强大的物理模型，目前只找到了一维和二维无磁场情况下的精确解，对二维有磁场的情况以及三维乃至 n 维的精确解的研究一直没有大的进展。现在随着计算机技术的发展，Ising 可以用计算机模拟，发展了蒙特卡罗算法等计算方法；虽然计算机只能模拟有限原子数的体系，但是这些模拟的结果也可以反映一些物理问题。本文将会讨论 n 维 Ising 模型的一般的蒙特卡罗模拟的矩阵法，并讨论一、二和三维情况的相变，以及考虑次近邻的情况。
**关键字**：n 维 Ising 模型　蒙特卡罗模拟　相变 矩阵法

## 一、研究背景介绍：

Ising模型是1920年由物理学家Wilhelm Lenz提出，并把这个问题交给了他的学生 Ernst Ising。具体的模型是这样的：

*考虑一个格点的集合，任意一个格点只能取两的状态{+1，-1}，对应于自旋向上和向下；而体系的能量由下式给出，*

$$E = -\sum_{<ij>} J s_i s_j - B\sum_i s_i \quad (1)$$

*其中<i j>表示对所有的相邻的原子对求和，J 大于零表示铁磁性，小于零表示反铁磁性而等于零表示无相互作用；B 是磁场强度，*

Ernst Ising 后来解出了一维 Ising 模型的严格解，因此这个模型用 Ising 的名字命名，但是结果是一维的情况下 Ising 模型无法反应相变的过程。至于二维和三维情况的求解大大的超出了 Ising 的能力范围；他也断言二维和三维都没有相变。

科学家对伊辛模型感兴趣的主要原因就是它能很好地显示连续相变过程 ,特别是在相变的临界温度附近的临界现象。 1907 年 Weis 提出了铁磁顺磁相变的分子场理论 ,也称为平均场理论。平均场理论的精神就是将一个自旋周围的所有自旋对它的作用平均成一个有效磁场。铁磁 顺磁相变的平均场理论与范德瓦尔斯的气液相变理论又是相对应的。由于四维及更高维空间的关联非常之强大 ,使其效应可以用一个平均的有效磁场来描述。但对于低维空间,平均场理论仅给出近似的定性结果 ,可以说是一个零级近似。为了获得更多的信息，许多科学家尝试着对平均场理论进行修改；但是这些小修小补，没有取得突破性的进展；必须要得到 Ising 模型的精确解！

1944 年耶鲁大学化学系Onsager教授用代数法求出了二维长方伊辛模型无磁场情况下的配分函数和比热的精确解【1】。算是对 Ising 研究的巨大的突破，但是具有次紧邻相互作用的二维伊辛模型和磁场下的二维伊辛模型至今没有精确解，更不要说三维了。

发展到了今天，人类开始求助于计算机解决问题，发明了许多的计算机模拟的算法，例如蒙特卡罗算法。虽然我们不知道解析表达式，但是数值结果也可以用于解决实际问题。就 Ising 模型而言，求解析解实在是过于困难，而 Ising 模型实在是一个重要的物理模型；数值求解就是解决 Ising 模型的不二之选。

## 二、关于解析解的讨论

首先必须申明本文中所有的讨论都是建立在周期性边界条件和正方格子的基础上的。前文已经提到了 Ising 解出了一维 Ising 的精确解，他使用的是组合法即用排列组合的方式直接地解出不同能量的简并度，进而求得配分函数。现在人们经常用到的是传递矩阵法。

设配分函数为 Z，根据定义有

$$Z=\sum_i e^{-\beta E_i},i\text{表示系统的一个微观态}\quad(2)$$

现在我们把（1）该写成如下的形式：

$$E=-\sum_{<ij>}Js_is_j-\frac{1}{2d}B\sum_{<ij>}(s_i+s_j)\quad(3)$$

其中 d 表示系统的维数。定义传递矩阵

$$T_{s,s'}=e^{\beta(Jss'+\frac{1}{2d}B(s+s'))}\quad(4)$$

对于 Z 就可以表示为

$$Z=\sum_{s_1}\sum_{s_2}\sum_{s_3}、、、、、\sum_{s_N}\prod_{<ij>}T_{s_is_j}\quad(5)$$

对于一维这种特殊情况，

$$Z=\sum_{s_1}\sum_{s_2}\sum_{s_3}、、、、、\sum_{s_N}T_{s_1s_2}T_{s_2s_3}\cdots T_{s_Ns_1}$$
$$=\mathrm{Tr}(\mathrm{T}^N)\quad(6)$$

但是对于一般的情况而言，上述方法是失效的。纵然任意维情况下（5）也可以改写成如（6）式那样矩阵的下标相邻相同似乎可以求和的形式（在周期性边界条件下，每个原子相邻原子有 2d 个，根据欧拉关于一笔画的条件【2】，可以找到一条曲线把每个原子都经过 2d 次），但是同一个下标 $s_i$ 会出现 2d 次，因此无法简单地去掉哑指标。

虽然用（5）无法给出任一情况的精确解，但是可以得到一个结论：把一个 2d*N 个原子的一维 Ising 模型，以一定的（由拓扑决定）的顺序令每 2d 个原子全同，就可以得到一个 d 维 N 个原子的 Ising 模型。

二维 Ising 模型的解法较为复杂，这里只讨论一下相应的结论。二维 Ising 模型（无磁场正方格子）在温度 T 下的平均能量由下式给出：

$$\bar{\mathrm{E}}=-2\mathrm{NJtanh}2\beta\mathrm{J}-\mathrm{NJ}\frac{\sinh^2 2\beta J-1}{\sinh 2\beta J\cdot\cosh 2\beta J}[\frac{2}{\pi}\int_0^{\frac{\pi}{2}}\frac{d\phi}{\sqrt{1-\kappa^2\sin\phi}}-1],\kappa=2\frac{\sinh 2\beta J}{(\cosh 2\beta J)^2}\quad(7)$$

二维 Ising 模型是允许相变的，临界温度由下式给出，

$$\sinh\frac{2J}{k_BT_c}=1 \quad (8)$$

如果近似的有，

$$\frac{k_BT_c}{J}\approx 2.269 \quad (9)$$

这一结论将会在下文的程序模拟中得到验证。

**三、n 维 Ising 模型的蒙特卡罗模拟**

1、关于 Ising 的蒙特卡罗算法【3】

考虑由 N 个原子组成的系统，每个原子都固定在格点上，自选只有经典的+1 或者-1 两种取值；我们只计算自旋间的相互作用的能量，和自旋态在磁场下的势能。首先，把系统的初态设置为所有的原子都自旋向上并给出磁场和温度的大小。再取出第一个原子，计算这个院子自选翻转需要的能量，记为$E_f$，如果$E_f$小于零，那么翻转这个原子的自旋；如果$E_f$大于零，使用计算机的相关指令集，例如 Python 的 numpy.random 产生一个零到一的随机数 r。如果

$$r\le e^{-E_f/k_BT},k_B是玻尔兹曼常数 \quad （10）$$

翻转这个院子的自旋。反之，保留原来的自旋方向。然后再取一个原子重复上述操作、、、直到把所有的原子循环一遍。理论上只要循环足够多的次数，这个系统最终的状态是符合一定温度下波尔兹曼分布的。

这种方法简单可行，而且可以适用于任意的情况下的 Ising 模型。只要找到一个统一的方法计算$E_f$就得到了一种数值模拟任意情况的 Ising 模型的方法了。

2、关联矩阵及其计算机生成方法

现在我们用态矢量来描述一个 Ising 模型的所有原子的状态，

$$|S\rangle=(s_1,s_2,、\ 、\ 、,s_N) \quad (11)$$

其中$s_i$表示第 i 个原子的状态；由于我们这里只考虑经典的情况，因此$s_i$只能取+1 或者-1 两个值。由于 Ising 模型中的原子是抽象的点，而分离的点的集合总是可以用一条曲线连起来的，也就是说任意整数维的 Ising 模型的原子都可以有一套一维坐标，正如（）中的 1 到 N。

如何在这样一套一维的坐标下表征不同原子的自旋间的相互作用呢？这里我们引入一个线性代数中的关联矩阵的概念。这个关联矩阵是一个 NxN 的矩阵，下文中用 A 表示，如

果 1 号原子和 2 号原子互为最近邻原子，那么 A[1,2]=A[2,1]=1;次近邻以及再次近邻，可以根据需要设置不同的值，但是对于一般的只考虑最近邻的原子的相互作用的 Ising 模型而言，我们都令他们为 0；对角项 A[i,i]在 A 矩阵中是为 0.那么能量就可以表示为以下的形式：

$$E=-\frac{1}{2}J\left\langle S\left|\mathrm{A}\right|S\right\rangle-\left\langle B\left|S\right.\right\rangle,\left\langle B\right|=(B_1,B_2,B_3、、、、、B_N) \quad (12)$$

式（）第一项中的 0.5 是由于对同一对原子重复计算而引入的修正。虽然如（）所示的形式，不方便代数求解，但是就数值求解而言，完成了从 n 维到 1 维的降维，实现了任意维 Ising 模型 E 的表达式形式上的统一。而且可以做考虑到非最近邻原子的作用，任意恒定磁场，任意形状等极端复杂情况的模拟。剩下来的问题只是线性代数。

当然了，对于任意情况的 A 是无法通过逻辑推导给出的，只能直接赋值。然而一些规则的情况，例如 n 维的正方或者六方，可以编写适当的程序，直接通过计算机生成。本文只涉及正方格子取周期性边界条件的情况，相关的程序见附录。主要的思想是：

令第 i 个原子的一维坐标 $N_i$ 和 d 维坐标 $(n_i^1,n_i^2,、、、,n_i^d)$ 的对应关系由下式给出

$$N_i=\sum_j n_i^j * n^{d-j} \quad (13)$$

其中总原子数 N 满足 $N=n^d$ ()。在定义原子间的距离 D 为

$$D_{ij}=\sum_k R(\left|n_i^k-n_j^k\right|) \quad (14)$$

$$\mathrm{R(x)}=\begin{cases} x, x<\dfrac{n}{2} \\ n-x, x>\dfrac{n}{2} \end{cases},(15)$$

式中的 R（x）是一个用于周期性边界条件修正的函数。如果不取周期性边界条件，则 R（x）=x。显而易见，对于最近邻原子 D=1,次近邻原子 D=2，以此类推。这样用两个循环，就可以生成需要的矩阵 A。

下式是用计算机生成的 3x3 的点阵，在考虑次近邻时（不妨设次近邻原子对应的矩阵元为 0.3）的关联矩阵：

```
A = {[0, 1.0, 1.0, 1.0, 0.3, 0.3, 1.0, 0.3, 0.3],
     [1.0, 0, 1.0, 0.3, 1.0, 0.3, 0.3, 1.0, 0.3],
     [1.0, 1.0, 0, 0.3, 0.3, 1.0, 0.3, 0.3, 1.0],
     [1.0, 0.3, 0.3, 0, 1.0, 1.0, 1.0, 0.3, 0.3],
     [0.3, 1.0, 0.3, 1.0, 0, 1.0, 0.3, 1.0, 0.3],
     [0.3, 0.3, 1.0, 1.0, 1.0, 0, 0.3, 0.3, 1.0],
     [1.0, 0.3, 0.3, 1.0, 0.3, 0.3, 0, 1.0, 1.0],
     [0.3, 1.0, 0.3, 0.3, 1.0, 0.3, 1.0, 0, 1.0],
     [0.3, 0.3, 1.0, 0.3, 0.3, 1.0, 1.0, 1.0, 0]}
```

## 四、Ising 模型的相变问题的程序模拟

1、一维 Ising 模型无相变的验证

根据（6）可求得当外磁场为零时，

$$Z = 2(2\cosh \beta J)^{N-1} \quad (16)$$

因为 Z 不显含 B，所以

$$M(B=0)=0 \quad (17)$$

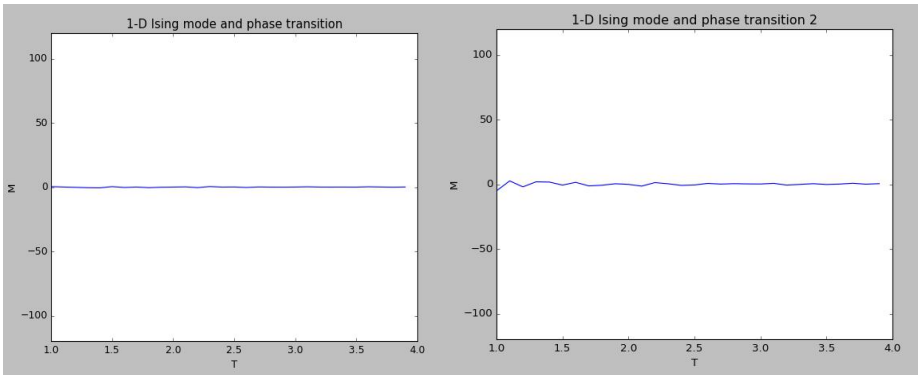
即一维 Ising 模型不存在铁磁相变。我们可以通过计算机模拟验证这个结论，如图 1 所示（100x1）。但是如果我们计算了次近邻的原子的相互作用，一维 Ising 模型能不能描述相变？这种情况也是没有解析解的。首先自旋与自旋的相互作用类似于磁偶极矩之间的相互作用，因此自旋激发的磁场 B 是反比于距离 r 的三次方的；由磁偶极矩在 B 场下的相互作用能可以表示为

E=Bs, s表示偶极矩（18）

综上如果我们认为如果近邻原子的 J=1，那么次近邻原子之间的 J=0.35。

模拟的结果如图 2 所示（100x1），考虑了次近邻的一维 Ising 模型还是无法描述相变过程。T 较小时的波动可能是统计涨落，无法说明什么。

图 1 ：　　图 2：




2、Ising 模型的临界点的拟合和次近邻飘移现象

众所周知，二维 Ising 模型及其更高维 Ising 模型可以描述相变，二维情况的相变临界点由（9）给出，但是高维的相变临界点现在还无法根据解析解直接得到。也是无法实验测量的，Ising 模型是一种理想的物理模型。这种情况下，计算机模拟就是唯一的可行的方案。

通过计算机模拟，我们验证了二维 Ising 模型正方格子（10x10）在无磁场条件下的临界点，如图 3 所示。计算机拟合的曲线并不是理想的在临界点出斜率趋于无穷大，但是从

图 3 中还是可以看出来在相变发生在 2.0 到 2.3 之间，平均 2.25 左右。我们也通过这一方式模拟了三维 Ising 模型（5x5x5）的相变,如图 5 所示。可以看出三维 Ising 模型（5x5x5）的临界点大概是 4.0.

图 3

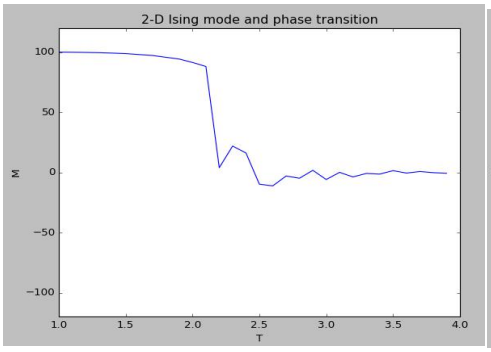



图 4

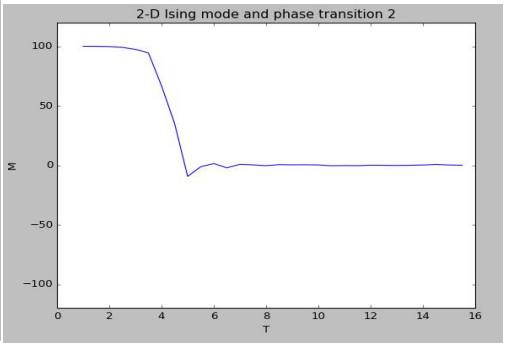



图 5

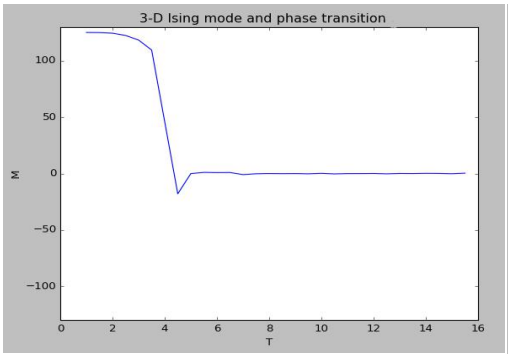
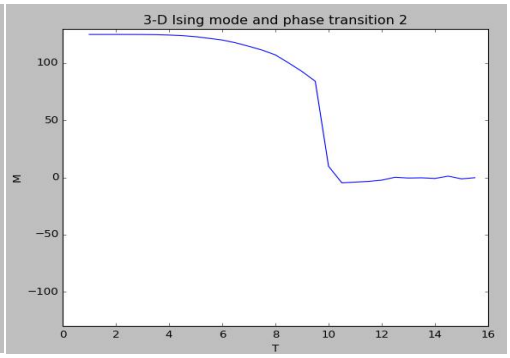


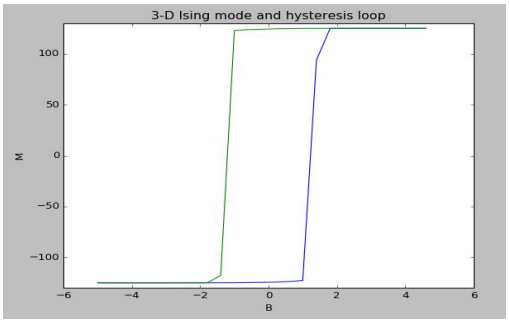

图 6




通过模拟计算了次近邻相互作用的二维（10x10）和三维(5x5x5)的相变，分别如图 4 和图 6 所示。发现次近邻的作用实际上是不可忽略的，会使临界点向 T 增大的方向移动接约一倍的距离。

3、三维 Ising 模型的磁滞回线模拟

Ising 模型的最初只是为了研究铁磁性而提出的,铁磁性的物质有一个明显的特征就是磁滞回线；磁滞回线允许铁磁性物质再退磁之后仍然保留磁性。这里我们选择最接近实际物质的三维 Ising 模型做磁滞回线的研究。不妨设温度为 2，那么 5x5x5 的三维 Ising 模型的磁滞回线由图 7 给出。图 7：



可以看出三维 Ising 模型是可以描述磁滞回线这种现象的，而且曲线的形态和实验测量得到的磁滞回线极其相似。

## 五、总结

Ising 模型是一个简单但是极其强大物理模型，但是当代还没有能解出任意维任意情况的 Ising 模型的配分函数的方法。蒙特卡罗模拟提供了一个现实可靠的方法让我们能用 Ising 模型研究实际物理问题，甚至可发现一些还没有用解析法还得到的结论。例如本文中的一维 Ising 模型在计算次近邻的情况下还是没有相变，如果考虑次近邻相互作用临界点会向 T 增大的方向移动接近临界点到原点的距离一倍的距离和三维 Ising 模型可以描述磁滞回线等等。

## Monte Carlo simulation of Ising mode

Neo Lee

**Abstract:**Ising mode is one of the simpliest Physical mode, but can well describle a number of physical processes, such as ferromagnetic phase transition, the alloy from order to disorder transition, liquid helium from normal state to super-conductive state and etc. Unfortunatly, up to now, we just find out the exact solutions of 1-D Ising mode and 2-D Ising mode without magnetic field and the study to solve the other cases of Ising mode is always having no result. Now computer offers us a new way to study Ising mode even if we don`t know the exact solutions. In this paper, the general Monte Carlo simulation to n-D Ising mode and the ferromagnetic phase transition of 1-D,2-D and 3-D cases will be discussed.

**Keywords:** Ising mode　　Monte Carlo simulation
Matrix method　　phase transition

Reference:

【1】Statistical and Thermal Physics With Computer Application　Harvey Gould
【2】An Eulerian gyrokinetic-Maxwell solver　J Candy,R E Waltz
【3】Computational Physics　Nicholas J.Giordano Hisao Nakanishi

## 附页：原始代码（Phython）

```
from math import *
import matplotlib.pyplot as p
import numpy.random as ran
#建立计算 Ising 模型的类
class Ising():
    def __inti__(self,J,N,T,istate):#初始化
        self.N=N
        self.T=T
        self.J=-J
        self.state=istate
        self.B=[0 for i in range(N)]
        self.Interactionmatrics=[[0 for i in range(N)] for i in range(N)]
        return None
    def setinteractionmatrics(self,newmatric):#设置相互作用矩阵或关联矩阵
        self.Interactionmatrics=newmatric
        return None
    def setB(self,B):#设置磁场
        for i in range(self.N):
            self.B[i]=-B
        return None
    def setT(self,T):#设置温度
        self.T=T
        return None
    def energy(self):#计算总能量
        E=0
        for i in range(self.N):
            for j in range(self.N):
                if self.state[i]==0 or self.Interactionmatrics[i][j]==0 or self.state[j]==0:
                    continue
                E=E+self.state[i]*self.Interactionmatrics[i][j]*self.state[j]*self.J*0.5
            if self.B[i]==0:
                continue
            else:
                E=E+self.B[i]*self.state[i]
        return E
    def M(self):#计算总磁矩
        M=0
        for i in range(N):
           M=self.state[i]+M
        return M
    def sweep(self):#一次演化
        for i in range(self.N):
            E1=self.nearbyinteraction(i)
```

```
            self.state[i]=0-self.state[i]
            E2=self.nearbyinteraction(i)
            e=E1-E2
            if e <0:
               p=exp(e/self.T)
               r=ran.random()
               if r>p:
                   self.state[i]=0-self.state[i]
        return None
    def nearbyinteraction(self,x):#计算近邻或次近邻原子间的相互作用能
        e=0
        for i in range(self.N):
            if self.Interactionmatrics[x][i]==0 or self.state[i]==0:
                continue
            e=e+self.state[x]*self.Interactionmatrics[x][i]*self.state[i]*self.J
        if self.B[x]!=0:
            e=e+self.B[x]*self.state[x]
        return e
#起辅助功能的函数
def oneton(x,n,k):#一维坐标到 n 为坐标的变化
    y=[]
    for i in range(k):
        y.append(int(x/n**(k-1-i)))
        x=x-y[-1]*n**(k-1-i)
    return y
def ntoone(y,n,k):#n 维坐标到一维坐标的变换
    x=0
    for i in range(k):
       x=x+n**(k-1-i)*y[i]
    return x
def R(x,n):#周期性边界条件修正函数
    if x>n/2:
        return n-x
    else:
        return x
def isnear(y1,y2,n):#计算距离
    k=len(y1)
    y=[R(abs(y1[i]-y2[i]),n) for i in range(k)]
    d=0
    for i in range(k):
        d=y[i]+d
    return d
def standarymetrics(n,k):#生成 n 维正方格子的关联矩阵或相互作用矩阵
    N=n**k
```

```
    metric=[[0 for i in range(N)] for i in range(N)]
    for x in range(N):
        for y in range(N):
            d=isnear(oneton(x,n,k),oneton(y,n,k),n)
            if d==1:
                metric[x][y]=1.0
            if d==2:#只记最近邻了略去
                metric[x][y]=0.35
    return  metric
def ave(x):#计算平均值
    l=len(x)
    X=0
    for i in range(l):
        X=X+x[i]
    avex=X/l
    return avex
#计算磁滞回线的主程序
line=Ising()
n=input('n=')
k=input('k=')
N=n**k
istate=[1.0 for i in range(N)]
line.__inti__(1.0,N,2,istate)
metric=standarymetrics(n,k)
line.setinteractionmatrics(metric)
B=[-5.0+0.4*i for i  in range(25)]
t=[ i for i in range(1000)]
a1=[]
a2=[]
M=[]
T=[0.5*i+1.0 for i in range(30)]
for j in range(50):
  if j<25:
    line.setB(B[j])
    M=[]
    for i in range(1000):
       M.append(line.M())
       line.sweep()
    a1.append(ave(M))
  else:
     line.setB(B[24-j])
     M=[]
     for i in range(1000):
        M.append(line.M())
```

```
            line.sweep()
        a2.append(ave(M))
a2.reverse()
p.plot(B,a1,B,a2)
p.ylim(-130,130)
p.xlabel('B')
p.ylabel('M')
p.title('3-D Ising mode and hysteresis loop')


#计算铁磁相变的主程序
line=Ising()
n=input('n=')
k=input('k=')
N=n**k
istate=[1.0 for i in range(N)]
line.__inti__(1.0,N,2,istate)
metric=standarymetrics(n,k)
line.setinteractionmatrics(metric)
B=[-5.0+0.4*i for i  in range(25)]
t=[ i for i in range(1000)]
a1=[]
a2=[]
M=[]
T=[0.5*i+1.0 for i in range(30)]
for j in range(30):
  line.setT(T[j])
  M=[]
  for i in range(1000):
      M.append(line.M())
      line.sweep()
  a.append(ave(M))
print(a)
p.ylim(-120,120)
p.plot(T,a)
p.xlabel('T')
p.ylabel('M')
p.title('1-D Ising mode and phase transition')#根据需要更改
```