MODULE: PHYS3190

Introduction to Monte Carlo Simulations

Dr Stefan Auer

School of Chemistry, University of Leeds, Leeds LS2, 9JT, UK.

s.auer@leeds.ac.uk, (office 1.12)

Lecture 1: Recap Statistical Mechanics

Introduction

Statistical Mechanics In your statistical mechanics lectures you have learned that thermodynamics can be derived from a knowledge of energy levels available to a system of molecules or even from the energy levels of individual molecules if the molecules do not interact.

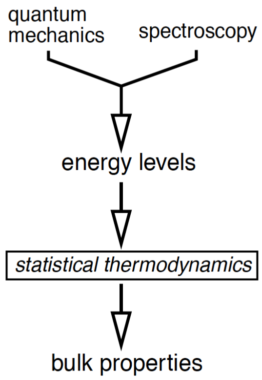

Figure 1. Statistical mechanics is a theoretical framework to compute the bulk properties of many-particle systems based on energy levels.

As illustrated in Figure 1, starting from the energy levels determined by quantum mechanics or measured experimentally by spectroscopy methods, in statistical mechanics we derived expressions for thermodynamic quantities by considering what happens when a large number of molecules explore the levels. The approach is therefore a statistical one and allowed us to calculate things like entropy, heat capacity and equilibrium constants from first principles.

The Canonical Distribution

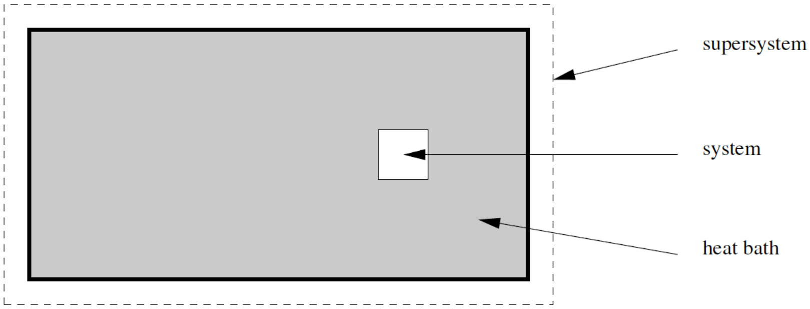

Figure 2. Illustration of a thermodynamic system composed of *N* molecules which are confined in a volume *V* and in contact with a heat bath of temperature *T*.

Central quantities in statistical mechanics are the canonical distribution and the canonical partition function. They relate to a thermodynamic system composed of *N* molecules that are confined in a volume *V*. The system is in contact with a heat bath of temperature *T* (see Figure 2), which implies that the internal energy of the system fluctuates. The probability to observe the system in a particular microstate *i* with energy E_i is given by the so-called canonical distribution function

$$p_i = e^{-E_i/kT} / Q_N \tag{1}$$

The denominator is the canonical partition function defined as the sum over all possible states

$$Q_N = \sum_j g_j e^{-E_j/kT} \tag{2}$$

where the energy of level *j* is E_j, its degeneracy is g_j and *k* is Boltzmann's constant. The summation forming the partition function covers all *j* values, and even if $j \to \infty$. The canonical distribution function is the equivalent of the well-known Boltzmann distribution, but for a system at constant temperature. It is important to note, that no assumptions about the nature of the system have been made in order to derive equation (1). In particular, there may be complicated interactions amongst the molecules in the system. The main leap made is supposing that the microstates of the system and their energies can be obtained.

Thermodynamic properties and the canonical distribution function The importance of the system partition function Q_N is that all thermodynamic properties can be derived from it. For example the internal (or mean energy *U*) of a system (which is a macroscopic quantity) composed of *N* atoms or molecules at a given temperature *T* and constant volume *V* can be calculated from the canonical partition function Q_N as

$$U = \langle E \rangle = \sum_i p_i E_i = \ldots = kT^2 \left(\frac{\partial \ln(Q_N)}{\partial T} \right)_V \tag{3}$$

Similarly, it can be shown that the Helmholtz energy is related to the system partition function by the *bridge relation*:

$$A = -kT \ln Q_N$$

and the entropy of the system is

$$S = k \ln Q_N + kT \left(\frac{\partial \ln Q_N}{\partial T} \right)_V$$

Thus, a central problem in statistical mechanics is to calculate Q_N, and only in some special cases the sum can be evaluated algebraically. The biggest specialisation one can make is that of non-interacting molecules. This allows to express the canonical partition function in terms of a molecular partition function, and furthermore we assumed that the energy of a molecule is given by the sum of energies due to translation, rotation, vibration, and electronic energy levels. However, computers open the way to consider more complicated systems (including interacting molecules). In this module we will discuss the Metropolis algorithm to calculate macroscopic properties of a system from the microscopic properties of the individual molecules.

Why is calculating the canonical partition function a challenge? In some special cases the sum can be evaluated algebraically. However, even if no algebraic solution is forthcoming, Q_N can be evaluated numerically because with increasing energy the exponential terms very rapidly become insignificant and hardly contribute to the sum. In practice therefore, the partition function summation has a limited rather than infinite number of terms. To appreciate this, if a harmonic oscillator has a frequency $\tilde{\nu} = 100$ cm^{-1} ($\nu = c\tilde{\nu}$ s^{-1}), comparing terms with n = 0 vs. n = 15 produces $e^{-h\nu/2kT} / e^{-31h\nu/2kT}$ ≈ 1330 at T = 300 K. This ratio increases dramatically as ν increases or T decreases. Next, consider the more involved case of calculating the partition function for the hindered rotation about bonds in an alkane that might be part of a polymer chain, or between adjacent residues in a protein. Imagine a polymer with 100 repeating units. Suppose that steric interaction between groups of atoms on adjacent units produces three potential energy wells and that there is only one energy level in each well. At any given time, the polymer chain can potentially exist in any one of $3^{100} \approx 10^{47}$ configurations. The partition function cannot be estimated by summation until all terms are added, because the sum does not tend to a limit after only a few terms but is complete only when all are included. To put 10^{47} configurations into context, there have only been ≈ 4×10^{17} s since the universe was created, and even if it were possible with a super-computer to calculate 10^{15} configurations each second, this single calculation would still take 10^{14} supercomputers the age of the universe to complete!

Monte Carlo Methods

Numerically solving integrals (such as the one to calculate Q_N in the classical limit) using a Monte Carlo method is done by repeatedly guessing the random values of one or more variables - such as variable x in function $f(x)$ - and assessing the result against some criteria. Consequently, the final answer is only an average of many of such guesses and is only an approximation to the true value. The Monte Carlo methods fall into two broad areas; the first is to use random numbers with a formula to perform integrations, and the second is to simulate physical processes at an elementary level, usually because these processes are too difficult to solve in any other way.

Integration

The direct Monte Carlo method is the simplest of all Monte Carlo methods to numerically integrate a function. The area corresponding to the integral is calculated by repeatedly guessing pairs of *x* and *y* values at random and evaluating the function $y = f(x)$ to see if y lies within the area bound by the integral or not, see figure 3. The ration of guesses inside/under curve to the total number of guesses is proportional to the integral. The more guesses that are made, the closer the answer becomes to the true value.

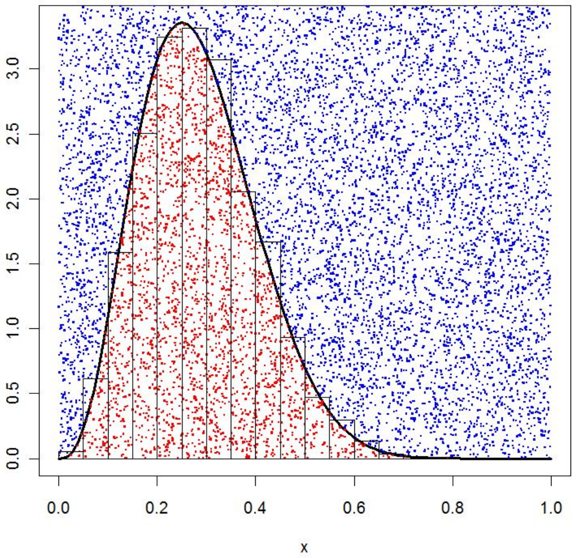

Figure 3. Illustration of the direct Monte Carlo method. The red points generated are guesses inside/under the area bound by the integral, the blue ones are guesses outside this area.

In general the integral is $Q = \int_a^b f(x)dx$. Two uniform distributed random numbers are chosen; one R_x between *a* and *b* and another R_y between limits *f*(*c*) and *f*(*d*), where the points *c* and *d* must be chosen to include the minimum and maximum of the function in the range *a* and *b*. A large number of pairs of points are generated, those for which $R_y \le f(R_x)$ are found and counted up. The integral is approximated as:

$$\int_a^b f(x)dx \approx \left(\frac{\text{\# guesses inside/under curve}}{\text{total \# guesses}} \right) \times A$$

where A is the integration area within all points fall (i.e. $A = (b-a)(f(d) - f(c))$ if $b > a$ and $f(d) > f(c)$).

Example code:

```
#!/usr/bin/env python2
# -*- coding: utf-8 -*-
"""
@author: Stefan
"""

import numpy as np #import numpy

def f(x): # function definition and parameters
    return x**2 # function body (in this case only the return value)

n = 2000 # number of guesses
xlim1 = 1.0 #x limit
xlim2 = 2.0 #x limit
ymax = f(2) #max y, min zero
s = 0 # initialize counter variable
A = (xlim2 - xlim1)*ymax # area A
for i in range(n): # for every value of n
    Rx = (xlim2-xlim1)*np.random.rand() + xlim1 # Set Rx, a random number
    Ry = ymax*np.random.rand() # Set Ry, a random number
    if Ry <= f(Rx): # if we are below the curve
        s = s+1 # increment counter

av_f = A*s/n # ratio below curve * area = integral estimate
```

Problem 1 (Workshop 1): Calculating the equation of state of real gases.

The virial coefficients are used in the description of real gases. The compression factor Z = 1 for an ideal gas, but is expanded as a series for a real gas,

$$Z = \frac{pV}{nRT} = 1 + B_2\left(\frac{n}{V}\right) + B_3\left(\frac{n}{V}\right)^2 + \ldots$$

The constants B_2, B_3, and so forth are the virial coefficients. The second virial coefficient B_2 can be related to the potential energy of interaction between molecules, which leads to non-ideal behaviour. The constant is

$$B_2 = 2\pi\int_0^\infty \left(1 - e^{-U(r)/kT}\right) r^2\, dr$$

where U is the interaction potential energy at the separation of r between molecules. In the case of a Lennard-Jones 6-12 potential

$$U(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right];$$

the integral has no analytic solution and has to be calculated numerically.

Using the direct Monte Carlo, calculate B_2 with the parameters for He, N_2, Ar, or Xe.

Strategy: The limits of the integration need to be addressed, because the limit of infinity is not generally possible with the Monte Carlo or other numerical methods. Additionally, the limit when $r = 0$ needs to be checked because here $U(0) = \infty$. In the later case the exponential term rescues the situation because $(1 - e^{-\infty}) = 0$, therefore the limit $r = 0$ is calculable. When $r = \infty$, U is zero, and the whole expression inside the integral is also zero. This can be seen by plotting $f(r) = \left(1 - e^{-U(r)/kT}\right) r^2$ and $U(r)$. In practice, a maximum value has to be put on r, and using the values given in the question, a plot indicates that $r = 20$ Å is quite sufficient. The function $f(r)$ is zero initially, and rises as r^2 for small r, however, this term is overwhelmed by the exponential at larger values of r.

Your tasks:

1. Plot $U(r)$ and $f(r)$ (0.5 mark)
2. Calculate B_2 using the direct Monte Carlo method (2 marks)
3. Check your result by using numerical integration routine of Python (1 mark)
4. Compare your result for B_2 to the literature. (0.5 mark)
5. What determines the accuracy of your result and why? Demonstrate understanding. (1 mark)
6. What is the molar volume of He(g) at 600 K and 600 bar according to the (a) ideal gas equation and (b) the virial equation? (1 mark)
7. Advanced question (possible to answer after lecture 2): Explain in your own words how it is possible to derive/obtain the expression for B_2 above from statistical mechanics? (1 mark)
8. Quality of presentation of report in Jupyter notebook (2 mark)
9. Bonus point: Justify why your work/report goes beyond just completing the tasks and deserves an extra mark (1 mark)

Submission of work:

- The work must be submitted as an ipynb on minerva
- The filename must be Surname-problem1.ipynb
- The self-assessment must be added at the end of the ipynb. Mark each task above.

Self-assessment:

- Aim is that you give an honest reflection on how you completed the tasks
- You will need to mark each task (for this problem tasks 1 to 9), and justify it when possible.
- Declaration of integrity (That the work you submitted is yours, and that you marked it fairly)

Deadline: 9th February 2026, 5pm

Comment:

- To compare effect of B_2 on ideal behaviour, the compressibility factor Z is often plotted versus p. Such plots can often be found in first year chemistry/physics text books, see e.g. Atkins "Physical Chemistry", chapter one on "Properties of Real Gases".

Table of molecules with corresponding parameter values used in Lennard-Jones potential

Gas atom/molecule	σ (in angstrom)	ε (in Joules)
He	2.56	$1.41 * 10^{-22}$
N_2	3.75	$1.32 * 10^{-21}$
Ar	3.4	$1.66 * 10^{-21}$
Xe	4.07	$3.04 * 10^{-21}$

About Python

Function Definition

```
def U(r):
    return 4*epsilon*((sigma/r)**(12)-(sigma/r)**(6))
```

Plots

```
import matplotlib.pyplot as plt
r = np.linspace(3.5, 10, 1000)
plt.plot(r, U(r))
```

Note: by default, Python will plot everything on the same graph. To clear the plot, use
plt.clear()

Integration

```
import scipy.integrate as integrate
B2N = integrate.quad(B2, 0, 20)[0]
```

Root finding

```
from scipy.optimize import broyden1
a=broyden1(B2, 4) # 4 is initial guess
```

Random number generation

```
import numpy as np
np.random.rand()
```