$\star$

FYS2160 - Thermal Physics

Oblig 01

Even Marius Nordhagen

September 14, 2016

1 Introduction

The purpose of this problem set is to learn making simplified models of the Einstein crystal and the spin system. First we are finding the multiplicity of each macrostate, just to find which macrostate that is the most probable, both analytical and numerical. We are also finding expressions for the entropy by using Boltzmann's formula, and from that we can determine the temperature.

Sometimes the mathematics is taking the attention from the underlying physics, but that is not good when you are writing a physic rapport. Therefore I will try to keep focus on the formalism and the physic aspects, and just use the mathematics as a tool.

All the script are coded in Python, and they can be found in the appendices in the back of this problem set.

2 Part I

In Part I we take a closer look at the Einstein crystal (also called an ideal crystal) and how energy can move when we have one and two subsystems. We are making some assumptions which simplify the reality a lot, but even though these seems to be rough, they normally fit well with the exact solution.

a)

So, we have 3 different oscillators with total energy 3. The possible combinations should be:

N_x	N_y	N_z
3	0	0
0	3	0
0	0	3
2	1	0
2	0	1
1	2	0
0	2	1
1	0	2
0	1	2
1	1	1

Table 1: All microstates for a system with 3 oscillators and 3 units of energy.

b)

The general formula for the multiplicity of an Einstein solid with N oscillators and total energy q is

$$\Omega(N,q) = \binom{q+N-1}{q} = \frac{(q+N-1)!}{q!(N-1)!} \tag{1}$$

By inserting $N = 3$ and $q = 3$, I should get $\Omega(N,q) = 10$ if I did the previous exercise in the right way:

$$\Omega(3,3) = \frac{5!}{3!2!} = \frac{120}{12} = \underline{10}$$

Oh yeah!

c)

For a system with two subsystems where subsystem A has $N_A = 2$ oscillators and $q_A = 5$ energy units, and subsystem B has $N_B = 2$ oscillators and $q_B = 1$ energy unit, the possible microstates are:

N_{A1}	N_{A2}	N_{B1}	N_{B2}
5	0	1	0
5	0	0	1
0	5	1	0
0	5	0	1
4	1	1	0
4	1	0	1
1	4	1	0
1	4	0	1
3	2	1	0
3	2	0	1
2	3	1	0
2	3	0	1

Table 2: All microstates for a system with two subsystems, where $N_A = 2$, $N_B = 2$, $q_A = 5$ and $q_B = 1$.

d)

The number of possible values for q_A is equal to the number of macrostates, and there can be shown that this number is always $q + 1$ for a system of two oscillators (it does not depend on what N_A and N_B are). This means that in our case we have 7 possible values of q_A (and 7 possible values of q_B since $q = q_A + q_B$). The possible values are:

q_A	q_B
6	0
5	1
4	2
3	3
2	4
1	5
0	6

Table 3: All possible macrostates for a system of two subsystems where $q = 6$.

e)

The task of this exercise is to find all the possible microstates for each of the macrostates.

First I will do this analytical, and for that I will use the general formula for the multiplicity of an Einstein solid from Equation (1):

$$\boldsymbol{q_A = 6:}\ \Omega(2,6)\cdot\Omega(2,0) = \frac{7!}{6!}\cdot 1! = 7$$

$$\boldsymbol{q_A = 5:}\ \Omega(2,5)\cdot\Omega(2,1) = \frac{6!}{5!}\cdot 2! = 12$$

$$\boldsymbol{q_A = 4:}\ \Omega(2,4)\cdot\Omega(2,2) = \frac{5!}{4!}\cdot \frac{3!}{2!} = 15$$

$$\boldsymbol{q_A = 3:}\ \Omega(2,3)\cdot\Omega(2,3) = \frac{4!}{3!}\cdot \frac{4!}{3!} = 16$$

$$\boldsymbol{q_A = 2:}\ \Omega(2,2)\cdot\Omega(2,4) = \frac{3!}{2!}\cdot \frac{5!}{4!} = 15$$

$$\boldsymbol{q_A = 1:}\ \Omega(2,1)\cdot\Omega(2,5) = 2!\cdot \frac{6!}{5!} = 12$$

$$\boldsymbol{q_A = 0:}\ \Omega(2,0)\cdot\Omega(2,6) = 1!\cdot \frac{7!}{6!} = 7$$

I have also done this numerical, by making a program where I'm plotting a histogram with the number of microstates on the y-axis and the different macrostates on the x-axis. For our specific case the plot is like this (*see Figure 1*).

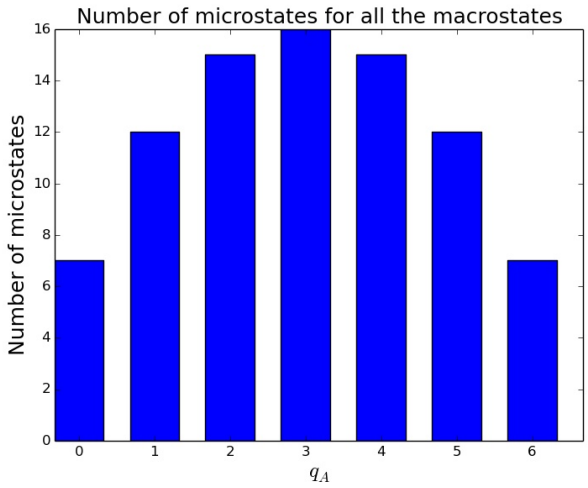

Figure 1: The probability of all the 7 macrostates where we have two subsystems and total energy of 6 units.

As we see, my numerical solution is equal to the analytical!

The total number of microstates can be found by adding the number of microstates for all the macrostates, or simply by $\Omega(4, 6)$. No matter which method we use, we get 84 possible microstates.

By dividing the number of possible microstates for a specific macrostate with the total number of microstates, we get the probability of the macrostate. This can easily be implemented in the program, so one can plot a histogram with the probability columns (*see Figure 2*).

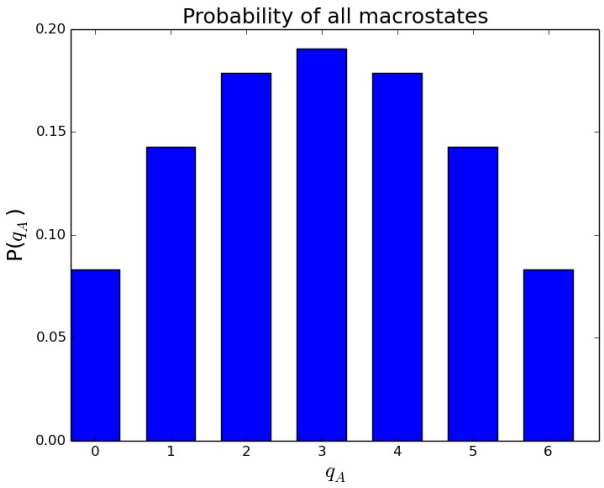

Figure 2: The number of microstates when we have two subsystems and total energy of 6 units.

My program, "oblig1e.py", can be found in Appendix A

f)

For the example where $N_A = N_B = 2$, $q_A = 5$ and $q_B = 1$, we saw that the number of microstates is 12 in the beginning, but it increases to 84 when the systems are set to thermal contact! The difference is of a factor 7, which is a big difference! This will happened in general, the number of microstates is always greater after thermal contact! Physical the reason is simple: the energy has many more ways to move after thermal contact, and that is what the microstates are intended to show.

g)

Below you can see the demanded plot (*Figure 3*).

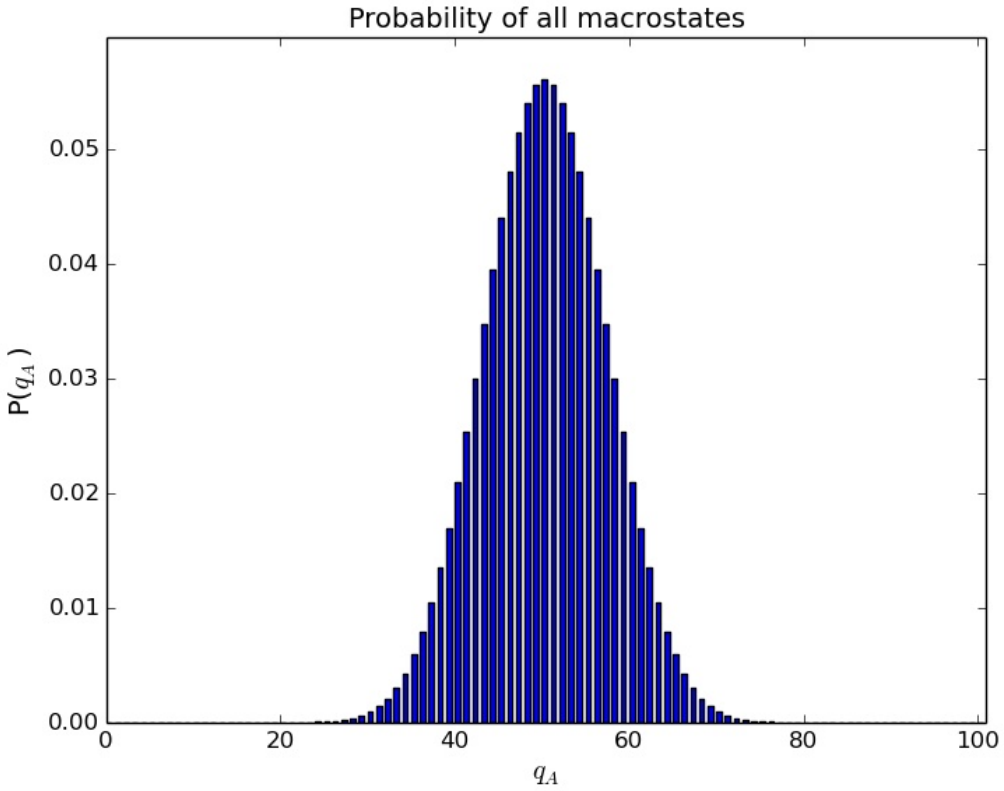

Figure 3: This figure shows the probability on the y-axis (heigth of the columns) and on the x-axis we have 101 columns since we have 101 macrostates

A fundamental law of thermal physics states that the most probable macrostate is when the systems are in equilibrium, and in our case this happens when $q_A = q_B = 50$. This is logical since we will have a Gaussian distribution which is max on the half of the interval ($f(x)$ is largest when $x = x_{max}/2$). The exact probability of this state could be found analytical by calculating $\Omega(100, 50)/\Omega(100, 100)$, but it will obtain a fraction with large factorials, so there is much faster to calculate it numerical. In my Python-script I have printed the probability to the terminal, and I find that it is approach

$$P(q_A = 50) \approx \underline{0.056}$$

I still use the program found in Appendix A, with just changing NA, NB and q.

h)

In this exercise I am going to show that the natural logarithm of the multiplicity can be simplified to

$$ln\Big(\Omega(N,q)\Big) = N\left(ln\frac{q}{N}+1\right)$$

If $N \gg 0$, one can make a really good approximation by doing a simplification of Equation (1):

$$\Omega(N,q) = \frac{(q+N-1)!}{(N-1)!} \approx \frac{(q+N)!}{(N)!}$$

since the ratio is almost equal. The next step is to take the natural logarithm of $\Omega(N,q)$:

$$ln\Omega = ln(q+N)! - lnq! - lnN!$$

Now I have to introduce Stirling's approximation to go future. A much used variant of Stirling's approximation states that

$$lnN! \approx NlnN - N \tag{2}$$

By using this on the expression above, I get

$$ln\Omega \approx (q+N)ln(q+N) - (q+N) - qlnq + q - NlnN + N = (q+N)ln(q+N) - qlnq - NlnN$$

To go even further I have to assume that $q \gg N$. I will try to simplify the first term:

$$ln(q+N) = ln\left[q\left(1+\frac{N}{q}\right)\right] = lnq + ln\left(1+\frac{N}{q}\right) \approx lnq + \frac{N}{q}$$

We can make these approximations since the first order Taylor expansion of $ln(1+x)$ gives us

$$ln(1+x) \approx x \tag{3}$$

if x is really small ($|x| \ll 1$). Finally we obtain that

$$ln\Omega \approx N\left(ln\frac{q}{N} + 1 + \frac{N}{q}\right) \approx \underline{N\left(ln\frac{q}{N}+1\right)}$$

we can make the last simplification by assuming that $q \gg N$.

i)

In general the entropy is given by

$$S \equiv kln\omega \tag{4}$$

So in our case we have

$$S \approx kN\left(ln\Big(\frac{q}{N}\Big)+1\right)$$

just by inserting the formula showed in the previous exercise into the definition of entropy. The reason why we use logarithm, is that q and N often are really large numbers.

j)

The first thing I want to do is to express S as function of E where S is the entropy and E is the total energy. For that I use the equation from the exercise description which connects q and E:

$$q = \frac{E}{\epsilon} \tag{5}$$

Then we obtain:

$$S = Nk\left(ln\left(\frac{E}{N\epsilon}\right) + 1\right) = NklnE - NklnN\epsilon + Nk$$

The reason I do this is because I already know the temperature formula, and it goes like

$$T \equiv \left(\frac{\partial S}{\partial E}\right)^{-1} \tag{6}$$

The formula is taken from the Lecture Notes[1] of this course. Since we are deriving with respect to the total energy, we are only interested in terms that include E. By inserting the entropy expression, we get

$$T = \left(\frac{\partial(NklnE)}{\partial E}\right)^{-1} = \left(Nk\frac{1}{E}\right)^{-1} = \frac{E}{Nk}$$

From this it follows that

$$E = \underline{NkT}$$

This a part of the equipartition theorem, which states that

$$E = \frac{1}{2}kT \cdot n$$

Where n is the number of free degrees (independent subsystems). Because the Einstein crystal has two free degrees for each oscillator, there is easy to see that these expressions are equal.

[1] "Elementary Thermal Physics using Python", Malthe-Sorenssen, Anders, (2016)

3 Part II

In Part II we study how energy is moving in a spin system (in practice an ideal gas). We are first counting the number of microstates, before we use this to find which macrostate that is most probable. The next step is to do some calculations to find three expressions for the multiplicity, and finally we are calculating the entropy and temperature. In conclusion much the same as we did in Part I, but now for an ideal gas.

k)

Since each of the N spins can have two possible states (+ or -), a N-spin system will have 2^N possible microstates. These we can enumerate by + or - for each N, but often we use $\uparrow$ for + and $\downarrow$ for -.

l)

Net spin is defined by $s = \frac{S_+ - S_-}{2}$.

From the exercise description we have that $E = -S\mu B$, so we can express the total energy as

$$U = -\mu B S_+ + \mu B S_- = -\mu B(S_+ - S_-) = \underline{-2s\mu B}$$

Where I have used the net spin formula.

m)

I decided to make a Python script which gives two plots. The first one shows the sum of the microstates. The sum is 0 if the number of spin up particles and equal to the number of spin down, i.e $S_+ = S_- = N/2$. It is positive when $S_+ > S_-$ and negative if $S_+ < S_-$. The plot looks like (*See Figure 4*):

The remaining plot is the histogram that was demanding in the exercise text (*see Figure 5*).

As we can see from the Figure, the distribution is approximately gaussian, but with some differences (for example is the middle column a little too high). These differences occurs because we are working with a finite number of random elements, which obviously will give some differences every time they are chosen. The more we increase the number of systems, the better is the gaussian approximation. If we increase the number of systems to infinity, we will have a perfect gaussian distribution. You may wonder why we have that distance between the columns, when the width is set to 0.90 (At least I wondered). The thing is that the energy formula never let us have an odd multiplied with μB ($S_+ - (N - S_+)$ can never be odd when N is even). Therefore we get quantified values.

The program, 'oblig1m.py', can be found in Appendix B

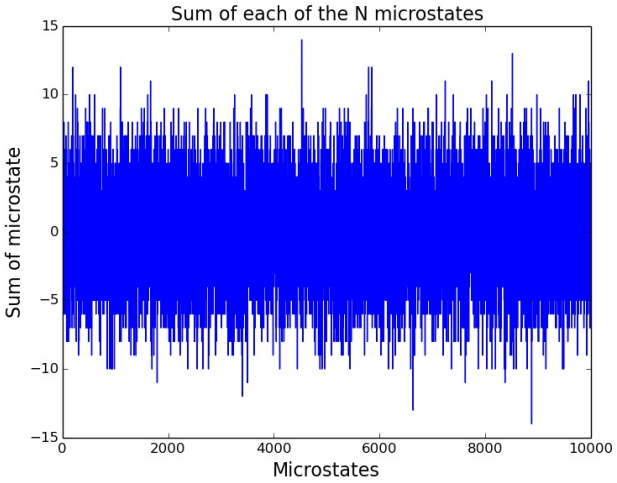

Figure 4: The sum of spin for all the 50 atoms in all the 10000 microstates

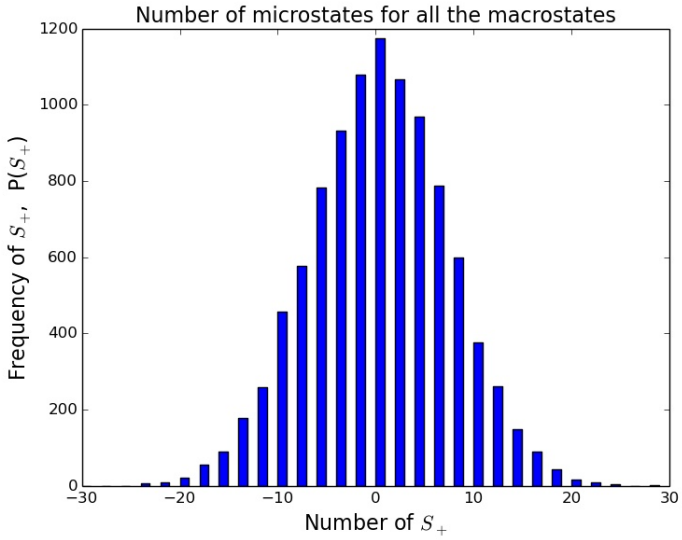

Figure 5: This figure shows the distribution of the sums of all the 10000 microstates

n)

The general formula for number of states is

$$\Omega(N, n) = \binom{N}{n} = \frac{N!}{n!(N-n)!}$$

By inserting into this formula and using that $N = S_+ + S_-$, we occur

$$\underline{\Omega(N, S_+) = \frac{N!}{S_+!S_-!}} \tag{7}$$

Because $N = S_+ + S_- \Rightarrow S_- = N - S_+$

o)

We have two connections between S_+ and S_-, and these are

$$2s = S_+ - S_-$$

$$N = S_+ + S_-$$

By mixing these formulae, one can show that

$$S_+ = \frac{N}{2} + s, \quad S_- = \frac{N}{2} - s$$

We can now insert this into Equation (7), and if we do so we get an expression for Ω which only depends on N and s:

$$\underline{\Omega(N, s) = \frac{N!}{(\frac{N}{2}+s)!(\frac{N}{2}-s)!}} \tag{8}$$

p)

One can now start with the expression in the last exercise to show that

$$\Omega(N, s) \approx \Omega(N, 0)e^{\frac{-2s^2}{N}}$$

Like the exercise text is hinting to, we have seen something similar in the lectures. On page 81 in the lecture notes, we are starting with a probability distribution on the form

$$P(N, u) = \frac{N!}{(N/2-u)!(N/2+u)!} \cdot 2^{-N}$$

Since the probability of a macrostate can be expressed as the multiplicity of the macrostate divided by the total multiplicity, and the total multiplicity is 2^N (as we argued for in exercise k)), this distribution is equal to the multiplicity formula in exercise o).

Equation (4.89) (from the lecture notes) states that

$$P(N,u) = C(N)exp\bigg(-\frac{u^2}{N/2}\bigg)$$

And because it is just a constant that makes the probability distribution different from the multiplicity, we can also write the formula for multiplicity as

$$\Omega(N,s) = C^*(N)exp\bigg(-\frac{u^2}{N/2}\bigg)$$

where $C^*(N) = C(N) \cdot 2^-N$ Actually we do not know anything about the constant $C(N)$, but we can find an expression for $C^*(N)$ by setting $s = 0$:

$$C^*(N) = \Omega(N,0)$$

Because $e^0 = 1$. So then we have shown that

$$\underline{\Omega(N,s) = \Omega(N,0)exp\bigg(-\frac{2s^2}{N}\bigg)}$$

With some help from the lecture notes.

Comment: We could obtain the same result by solving it analytical, but to get the right factor in the exponent, we need to do Taylor expansion to second order. This is much more work than using the results from the lecture notes.

q)

In this exercise I was asked to compare the the multiplicity expression from the previous exercise with the histogram from exercise m). That plot looks like this:

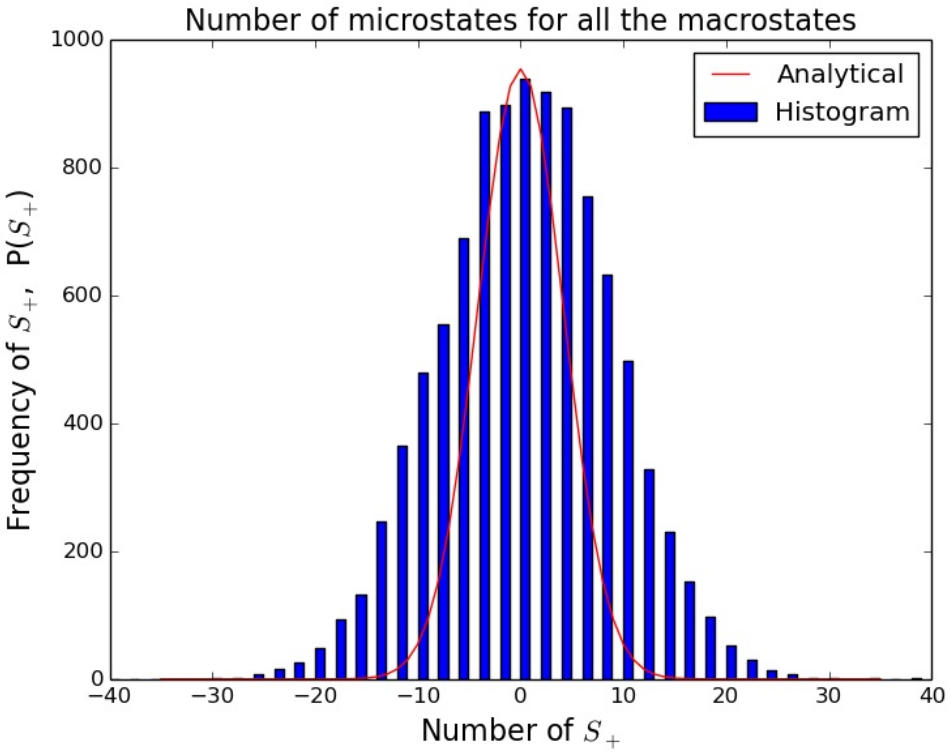

Figure 6: Here I have compared the analytical expression from exercise p) with a histogram of the distribution for the same values of N

My first try was not successful, the analytical solution was way too high! Then I realized that $\Omega(N, 0)$ will blow up the function up to the sky, so I decided to normalize the areal to 1. Statistical it will happened because it takes care of all the possible and impossible microstates, but we are just looking for M microstates.

Then I had to start on new. The idea is that I can calculate the area of the histogram, and multiply the analytical solution with it to get equal area. It makes the plot looking better, but it is still not a perfect plot (the width is too small). I do not know why the error occur.

You can find the program 'oblig1q.py' in Appendix C

r)

In this exercise I will go back to the exact multiplicity formula from *Equation 7*. This is necessary if we want to find exact formulae for entropy and temperature. Since the expression for entropy (*see Equation (4)*) depends on $ln(\Omega)$, an obvious procedure is to start finding an expression for that.

$$\begin{aligned} ln\Omega(N, S_+) &= ln\left(\frac{N!}{S_+!(N-S_+)!}\right) \\ &= lnN! - lnS_+! - ln(N-S_+)! \\ &\approx NlnN - N - S_+lnS_+ + S_+ - (N-S_+)ln(N-S_+) + N - S_+ \\ &= N\Big(lnN - ln(N-S_+)\Big) - S_+\Big(lnS_+ - ln(N-S_+)\Big) \\ &= Nln\left(\frac{N}{N-S_+}\right) - S_+ln\left(\frac{S_+}{N-S_+}\right) \end{aligned}$$

By inserting into the expression for entropy, we occur

$$\underline{S = k\left(Nln\left(\frac{N}{N-S_+}\right) - S_+ln\left(\frac{S_+}{N-S_+}\right)\right)}$$

s)

The last challenge is to find an expression for the temperature. Normally this would correspond to find an expression for T, but because the variable $\frac{1}{T}$ is more intuitive, we often find it more interesting. I have already found this formula in *Equation (6)*.

$$\frac{1}{T} = \left(\frac{\partial S}{\partial E}\right) = \left(\frac{\partial S}{\partial S_+}\right)\left(\frac{\partial S_+}{\partial E}\right)$$

Where I also have used the hint from the exercise text. As you can see, I need to express S with respect to S_+, and S_+ with respect to E. The expressions must be

1. $$S = k\left(Nln\left(\frac{N}{N-S_+}\right) - S_+ln\left(\frac{S_+}{N-S_+}\right)\right)$$

2. $$E = -2s\mu B = -(S_+ - S_-)\mu B = -S_+\mu B + (N-S_+)\mu B = -2S_+\mu B + N\mu B$$

So far so good. I will do the calculations in two steps

$$\frac{\partial S}{\partial S_+} = \frac{\partial}{\partial S_+}\left(k\left(Nln\left(\frac{N}{N-S_+}\right) - S_+ln\left(\frac{S_+}{N-S_+}\right)\right)\right)$$

To make the derivation correctly we have to be aware. I will split the calculation into two subcalculations just to keep the overview:

$$\frac{\partial}{\partial S_+}\left(kNln\left(\frac{N}{N-S_+}\right)\right) = kN\left(\frac{N-S_+}{N}\right)\left(\frac{N}{(N-S_+)^2}\right) = \frac{kN}{N-S_+}$$

$$\frac{\partial}{\partial S_+}\left(kS_+ln\left(\frac{S_+}{N-S_+}\right)\right) = kln\left(\frac{S_+}{N-S_+}\right) + kS_+\left(\frac{N-S_+}{S_+}\right)\left(\frac{N}{(N-S_+)^2}\right)$$
$$= kln\left(\frac{S_+}{N-S_+}\right) + \left(\frac{kN}{N-S_+}\right)$$

Here I have used the quotient rule to derive the fractions, and in total we get:

$$\frac{\partial S}{\partial S_+} = -kln\left(\frac{S_+}{N-S_+}\right) = kln\left(\frac{N-S_+}{S_+}\right)$$

The next step is to derive S_+ with respect to E, but since I already have an expression for E (with respect to S_+), I will take the inverse:

$$\frac{\partial S_+}{\partial E} = \left(\frac{\partial E}{\partial S_+}\right)^{-1} = \left(\frac{\partial}{\partial S_+}\Big(-2S_+\mu B + N\mu B\Big)\right)^{-1} = \Big(-2\mu B\Big)^{-1} = -\frac{1}{2\mu B}$$

Finally we obtain the expression for (inverse) temperature:

$$\underline{\frac{1}{T} = \frac{k}{2\mu B}ln\left(\frac{N}{S_+} - 1\right)}$$

4 Appendix A

Below you can find the code from 'oblig1e.py', that was used in exercise *e)* and *g)*.

Under the script you can find a description of the code.

```
from matplotlib import pyplot as plt
import numpy as np

NA=50
NB=50
q=100

#Factorial
def factorial(number):
    a=1
    for n in range(number):
        a=a*(n+1)
    return a

#Multiplicity of an Einstein solid
def Omega(N,q):
    numerator=factorial(q+N-1)
    denumerator=factorial(q)*factorial(N-1)
    return int(numerator/denumerator)

#Counting number of macro- and microstates
Y=[]
print 'For these choices of NA, NB and q we have %.f different
    macrostates'%(q+1)
for i in range(q+1):
    A=Omega(NA,i)
    B=Omega(NB,q-i)
    Y.append(float(A)*B/Omega(NA+NB,q))
    #print 'qA=%.f gives %.f different microstates'%(i,A*B)

#Plot
SZ={'size':'14'}              #Size of labels
width = .60                     #Width of columns
ind = np.arange(q+1)

plt.bar(ind, Y, width=width)    #Plotting histogram
#plt.xticks(ind + width / 2, ind) #Placing the columns above indexes
plt.title('Probability of all macrostates',**SZ)
plt.xlabel('$q_A$',**SZ)
plt.ylabel('P($q_A$)',**SZ)
plt.show()
```

Here I will try to explain what I am doing in this program. First I calculate the facto-

rial. I think this little script is simpler than importing factorial from math, but this is individual.

The next function is calculating the multiplicity of the Einstein crystal, which I call on in the for loop and storing the multiplicity of every single macrostate into a list. Then I make a histogram with the tools from matplotlib.

5 Appendix B

Below you can find the code from 'oblig1m.py', that was used in exercise *m)*.

Under the script you can find a description of the code.

```
import numpy as np
import matplotlib.pyplot as plt

N = 10000
M = 50

F = np.zeros([N,M])     # Nested list with M microstates for N systems
S = np.zeros(N)         # List with all sums. The sum is number of S+
P = np.zeros(M)         # List of sums sorted

for i in range(N):
    for j in range(M):
        F[i,j] = int(np.random.random()+0.5) # Filling F
    S[i] = np.sum(F[i,:])                    # Counting number of S+ in each
        state
    for k in range(M):
        if int(S[i]+0.5) == k:               # Sorting S-array
            P[k] += 1                        # Counting number of diff. type
    S[i]=S[i]-25

#Plot
SZ={'size':'16'}          #Size of labels

x=np.linspace(0,N,N)
plt.title('Sum of each of the N microstates',**SZ)
plt.xlabel('Microstates',**SZ)
plt.ylabel('Sum of microstate',**SZ)
plt.plot(x,S)
plt.show()

width = .9                    #Width of columns
ind = np.zeros(M)             #Array with number of columns
for i in range(M):
    ind[i]=(M-i)-i;
plt.bar(ind, P, width=width)
#plt.xticks(ind + width / 2, ind) #Placing the columns above indexes
```

```
plt.title('Number of microstates for all the macrostates',**SZ)
plt.xlabel('Number of $S_+$',**SZ)
plt.ylabel('Frequency of $S_+$, P($S_+$)', **SZ)
plt.show()
```

This program may looks some more ugly than the previous, so I will describe what it is doing. First I create some arrays with different length and shape, just so I can store numbers later. Then I am filling the nested array F with random numbers (either 0 or 1, to make the calculations some simpler. For example the sum of an array easily tells me the number of $1's$, or in practice S_+). I store the sums into an array S, counting the number of each sum in S, and storing the numbers of each sum into P-array. After that I plot the random numbers and the histogram of the macrostates.

6 Appendix C

Below you can find the program called 'oblig1q.py', which I used in exercise *q)*. The start of the program is similar with the program 'oblig1m.py', which you can find in the previous appendix. Even though I decided to put all the program here since there is some important differences in the main part of the program.

Under the script you can find a description of the code.

```
import numpy as np
import matplotlib.pyplot as plt

M = 10000
N = 70

F = np.zeros([M,N])     # Nested list with M microstates for N systems
S = np.zeros(M)         # List with all sums. The sum is number of S+
P = np.zeros(N)         # List of sums sorted

for i in range(M):
    for j in range(N):
        F[i,j] = int(np.random.random()+0.5) # Filling F
    S[i] = np.sum(F[i,:])                    # Counting number of S+ in each
    for k in range(N):                       # state
        if int(S[i]+0.5) == k:               # Sorting S-array
            P[k] += 1                        # Counting number of diff. type

def Omega(N,s):
    return np.exp(-(2*s**2)/N)              # Dropping (2**N), will norm
                                             # normalize later
slist=np.zeros(N+1)
sl=np.linspace(-N/2,N/2,N+1)
i=0
for s in sl:
```

```
    slist[i]=Omega(N,s)
    i+=1
slist=(slist/sum(slist))*M   #Makes the area of exact equal to
                             #the area of the histogram
#Plot
SZ={'size':'16'}          #Size of labels
width = .9                   #Width of columns
ind = np.zeros(N)            #Array with number of columns
for i in range(N):
    ind[i]=(N-i)-i
plt.plot(sl,slist,'-r')
plt.bar(ind, P, width=width)
#plt.xticks(ind + width / 2, ind) #Placing the columns above indexes
plt.title('Number of microstates for all the macrostates',**SZ)
plt.xlabel('Number of $S_+$',**SZ)
plt.ylabel('Frequency of $S_+$, P($S_+$)', **SZ)
plt.legend(['Analytical','Histogram'],loc='best')
plt.show()
```

As already said, the start of the program is similar to the previous code, so I will not describe this part (*see Appendix B*). After that I make a function which calculates the multiplicity with N and s as arguments. I decided to remove the constant 2^N, but this does not matter since I am normalizing later (2^N is a very large number when N is large). Then I have a for-loop that is calling on $Omega(N, s)$, with help from the list sl which stores all possible values of s.The loop also stores the different outcomes from $Omega(N, s)$ in an array, which is normalized by dividing on the sum of the array.

There can be shown that the total area of the histogram is equal to M. This is because each of the columns has width 1 and the total height of the columns is M. By multiply the normalized array with M, we therefore should get the same area for the histogram and the gaussian, but it does not looks like that. I have to apply that I have no idea what is wrong.

The last part consists of different plot commandos.