33-151 Matter & Interactions I Fall 2019

Name (Printed) ______________________ Section ______

Instructor Signature for 9.P59 (a-c): ______________________

Instructor Signature for 9.P60: ______________________

Due Date: Monday of Week 14.

Distribution of particles between two boxes

Introduction

In this exercise, we will numerically calculate the probability distributions of N particles distributed between two boxes. As we showed in class, the distribution is given by:

$$P(N_1, N_2) = \frac{N!}{N_1!N_2!}\left(\frac{V_1}{V}\right)^{N_1}\left(\frac{V_2}{V}\right)^{N_2} \tag{1}$$

with the condition that the total number of particles $N_1+N_2 = N$ is constant and $V_1+V_2 = V$. We will want to be able to vary the two volumes, keeping the total volume fixed. We also showed that give a probability function, $P(x)$, we can easlily compute the average of some quantity, y, as

$$< y > = \sum_i P(x_i)\, y_i \,,$$

where the sum is over all possible values of x (and y). For example,

$$< N_1 > = \sum_{N_1=0}^{N} P(N_1)\, N_1 \,.$$

In order to simplify things, we will choose our total volume to be $V = 1$, and will start with a total of $N = 10$ particles. However, we may want to increase N to a larger value later, so be sure to allow this flexibility in your program.

For the simulated experiment, you will have the computer throw N particles into the two subsystems with the appropriate probability. A random number between 0.0 and 1.0 can be generated by the random function as given above, no argument is needed.

Programming Assignment

We will be doing problems 9.P59 and 9.P60 from the textbook.

9.P59. In this problem, we will numerically calculate the probability distributions of N particles distributed between two boxes. These probabilities are given by equation 1 where we have the constraints that

$$\begin{aligned} N &= N_1 + N_2 \\ V &= V_1 + V_2 \,. \end{aligned}$$

As we write our program, we want to be able to easily change V_1 and V_2 subject to these constraints, but to start, choose that $V_1 = 0.5\,V$. (a) Write a program which will draw a plot of the probability distribution as a function of N_1 for N_1 goes from 0 to N. Include a calculation if the average value of N_1 based on your probability distribution. Have your program print out the average value of N_1, the most probable value of N_1 and the probability of the average value. (b) Modify your program to plot the natural logarithm of the probability as a function of N_1. Verify that the maximum of this plot is where the maximum was from part (a). (c) Use your program to carry out your calculations and make plots for $N = 50$ particles and $V_1 = 0.25V$, $0.5V$ and $0.75V$. Record the average and most probable values of N_1, for these cases.

What are the most probable values for the three different volumes?

Are these what you expect them to be?

9.P60. Write a program that randomly throws one hundred particles, one at a time, into two identical boxes. After the 100 particles have been thrown, record how many times we found $N_1 = 0, 1, 2, \cdots, 100$ in the first box. Make a plot of these agains the number N_1. Does the location and the width of the histogram distribution agree with what you expected from the theoretical probability distributions and the maximum entropy?

Does this randomly generated distribution look like what your analytic expression looks like?

Vary the number of experiments from a small number like 10 up to 1000 and comment on the agreement.

There is a program shell at the end of this document to get you started, but you will need to still do quite a bit of work to get things going. We have broken this exercise into two related programs. You are welcome to combine them all into one larger program if you want, but there are two signatures associated with this

Programing Hints

You are likely to need the factorial, binomial coefficient and natural log functions in VPython. Recall that the factorial of zero is defined to be one, $0! = 1$. The functions are given as:

$$\begin{aligned}
\text{factorial(N)} &= N! &&= N \cdot (N-1) \cdot (N-2) \cdots 2 \cdot 1 \\
\text{combin(a, b)} &= \left(\begin{array}{c} a \\ b \end{array}\right) &&= \frac{a!}{b!(a-b)!} \\
\log(x) &= \ln(x) &&= \log_e(x) \\
\text{random()} &= &&\text{random number from 0 to 1}
\end{aligned}$$

Note that in Equation 1, the ratio

$$\frac{N!}{N_1!N_2!} \quad = \quad \text{combin}(N, N_1)\,.$$

To raise a number to a power, we can do the following

$$a^N \quad = \quad a**N$$

We also note that because of internal precision of our computer, there is a maximum value for which we can compute the factorial. However, the upper limits for the binomial coefficients, combin(a, b) are larger as the algorithm does not fully evaluate the $a!$ term in the expression. Thus, if you can write things in terms of combin(a, b), you will be able to got to larger numbers. So rather than writing

```
#
N2 = N - N1
Prob = factorial(N)/(factorial(N1)*factorial(N2))
#
```

it is better to write the probability as

```
#
Prob = combin(N,N1)
#
```

which will not suffer from numerical problems.

In the random-selection program, 9.P71, you will create a histogram which is used to record the number of times each outcome occurs. There are some commented-out notes at the end of your shell on this. The trick is to build a list for each possible value of N_1 as seen in volume V_1, and repeat the experiment many times. Each time you test how many are in V_1, and then increment the appropriate element in the list. After you have done many numerical experiments, plot the number of times each value of N_1 occurred. The following hints might be helpful when you do the randomized throwing of the particles. You need to understand what this is doing before trying to code it up ....

```
#
# Set up an empty list to contain the results:
#
i=0
while (i < N):
    randomdata.append(0)
#
# Carry out the Nexpt numerical experiments, the index i
# represents a particular experiment.
#
Ne=0
while (Ne<Nexpt):
#
# Throw N total particles:
#
   N1 = 0
   Np = 0
   while(Np<N):
#
# decide if the particle went into V1...
#
        if random()<(V1/V):
            N1+=1
#
# get the next particle in this experiment
        Np+=1
#
# N1 is the number of times the particle went into the first
# volume in this experment. Add one to the appropriate element
# of our list.

    randomdata[N1]+=1.0
#
#  Go to the next experiment.
#
    Ne+=1
#
```

Program Shell

```
from math import *
from vpython import *
#
# Set up the total number of samples and experiments
```

```
#
N = 0        # Number of particles to distribute
V = 1.0      # Let the total volume be one unit
ratio = 0.   # Set up the ratio V1/V
#
# The following is an empty list that allows us to store the probability
# for each of the N1=0 to N1=N possibilities.
#
probdata=[]
#
# Set up the two volumes:
#
V1 = V*ratio
V2 = V-V1
#
# We also need to determine at which value of N1 we have the maximimum
# of the probabability, so in each step of our loop, we need to check
# if the current probability is max, and if so record the location (Max)
# and probability (MaxValue). We will also compute the average.
#
Max = 0      # Most probable value of N1
MaxValue = 0 # Probability of most probable value of N1
Average = 0  # Average value of N1
#
# Setup the graphical displays. You are likely to need to modify
# some of this as you go through the various programs.
#
scene1=canvas(title='Combinations',caption='Probability Distribution',center=vector(0,0,
#
s='<b>Distribuion of Particles in a Box</b>'
graph(title=s, xtitle='N', ytitle='Probability',xmin=0,xmax=N,
      ymin=0,width=400,height=200)
#
probplot=gvbars(color=color.blue)
#
# ---------------------------------------------------
# Initialization Complete, Start Actual Calculations
# ---------------------------------------------------
#
N1=0
while (N1<=N):
#
```

```
# Compute your probability for our choice of N1
#
    p = 0
#
# Add this probability to the list of probabilities
#
    probdata.append(p)
#
# Do a check to see if the current value of p is the maximum value.
# (Max and MaxValue)
#
#     if (p>MaxValue):
#
#
# Update your running sum for the average value of N1
# (Average)
#
    Average = 0
#
# Go to the next value of N1
#
    N1+=1
#
# We have finished the loop, print out the numerical results.
#
print (" Analytic Results:" )
print (" Average Value of N =",Average)
print (" Most Probable N =",Max,"; Probability=",MaxValue)
print (" --------------------")
#
# Plot the probability distributions in a window, assuming that we
# have the filled probdata array from above.
#
i=0
while (i<=N):
    probplot.plot(pos=(i,probdata[i]))
    i+=1
```