EE2703 : Applied Programming Lab
Assignment 6
Tubelight Simulation

V Nandha Varman
EE19B043

1 Outline:

In this assignment we will simulate a tubelight using a 1D model and plot the electron density, intensity of emitted light and the electron phase space.

A uniform electric field is present, that accelerates electrons. Electrons are emitted by the cathode with zero energy, and accelerate in this field. When they get beyond a threshold energy E_0, they can drive atoms to excited states. The relaxation of these atoms results in light emission. In our model, we will assume that the relaxation is immediate. The electron loses all its energy and the process starts again.

Electrons reaching the anode are absorbed and lost. Each "time step", an average of N electrons are introduced at the cathode. The actual number of electrons is determined by finding the integer part of a random number that is "normally distributed" with standard deviation of 2 and mean 5.

2 Code:

2.1 Parameter handling:

In this part we,

1. Import the required libraries
2. Set default values to parameters
3. Optionally, get parameter values through command line arguments
4. Define empty arrays and lists for use in the simulation

```
# importing necessary modules
import sys
from pylab import *

n = 100                     # Spatial grid size
M = 5                       # Number of electrons injected per turn
Msig = 2                    # Standard deviation of electrons injected per turn
nk = 500                    # Number of turns to simulate
u0 = 5                      # Threshold velocity
p = 0.25                    # Probability that ionisation will occur


if len(sys.argv) > 7:
    print("Error!! Too many command line arguments passed.")
    exit()

# getting parameter values from optional command line arguments
else:
    for i in range(len(sys.argv)):
        if i == 1:
            n = int(sys.argv[1])
        elif i == 2:
            M = int(sys.argv[2])
        elif i == 3:
            Msig = int(sys.argv[3])
        elif i == 4:
            nk = int(sys.argv[4])
        elif i == 5:
            u0 = int(sys.argv[5])
        elif i == 6:
            p = float(sys.argv[6])


# empty arrays to store electrons information
xx = zeros(n * M)           # Electron position
u = zeros(n * M)            # Electron velocity
dx = zeros(n * M)           # Displacement in current turn

# empty lists for intensity, position and velocity
I = []                      # Intensity of emitted light
X = []                      # Electron position
V = []                      # Electron velocity
```

2.2 Iteration:

In this part, we run a loop for nk turns. In each turn electrons are injected(whose number is given by a normal distribution). The electrons are accelerated by a fixed electric field, and of those which cross the threshold velocity a few are chosen by probablility p to have undergone ionization.

The location of ionization is approximated and is added to the I list. The velocity of these electrons is then reset to zero(inelastic collision).

The locations and velocites at the end of the turn is then added to the X and V lists.

```python
# getting the locations which are occupied by electrons
ii = where(xx > 0)[0]
# loop simulating tubelight operation
for i in range(nk):
    # updating electrons positions and velocity
    dx[ii] = u[ii] + 0.5
    xx[ii] += dx[ii]
    u[ii] += 1

    # setting the parameters of electrons that have hit the anode to zero
    hit_anode = where(xx[ii] > n)[0]
    xx[ii[hit_anode]] = 0
    u[ii[hit_anode]] = 0
    dx[ii[hit_anode]] = 0

    # finding the electrons that are ionized
    kk = where(u >= u0)[0]
    ll = where(rand(len(kk)) <= p)[0]
    kl = kk[ll]
    # resetting their velocities after ionizaiton to zero
    u[kl] = 0
    # assigning the position of the ionized electrons randomly between the previous
     ↪  and current xi
    rho = rand(len(kl))
    xx[kl] -= (dx[kl] * rho)
    # adding a photon at the location of electron excitaion
    I.extend(xx[kl].tolist())

    # determining the actual number of electrons injected
    m = int(randn() * Msig + M)
    # finding the unused indices
    not_ii = where(xx == 0)
    # finding the number of electrons to fill
    empty_slots = min(len(not_ii), m)
    # filling electrons in the unused slots
    xx[not_ii[:empty_slots]] = 1
    u[not_ii[:empty_slots]] = 0
    dx[not_ii[:empty_slots]] = 0

    # getting the locations which are occupied by electrons
    ii = where(xx > 0)[0]

    # adding location and velocity of electrons to the X and V list
    X.extend(xx.tolist())
    V.extend(u.tolist())
```

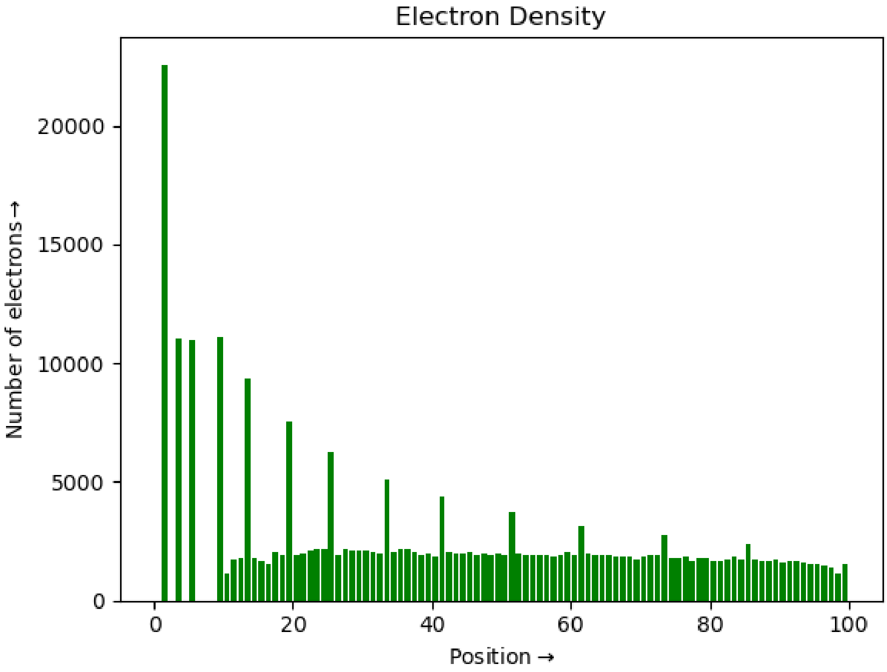
2.3 Plots:

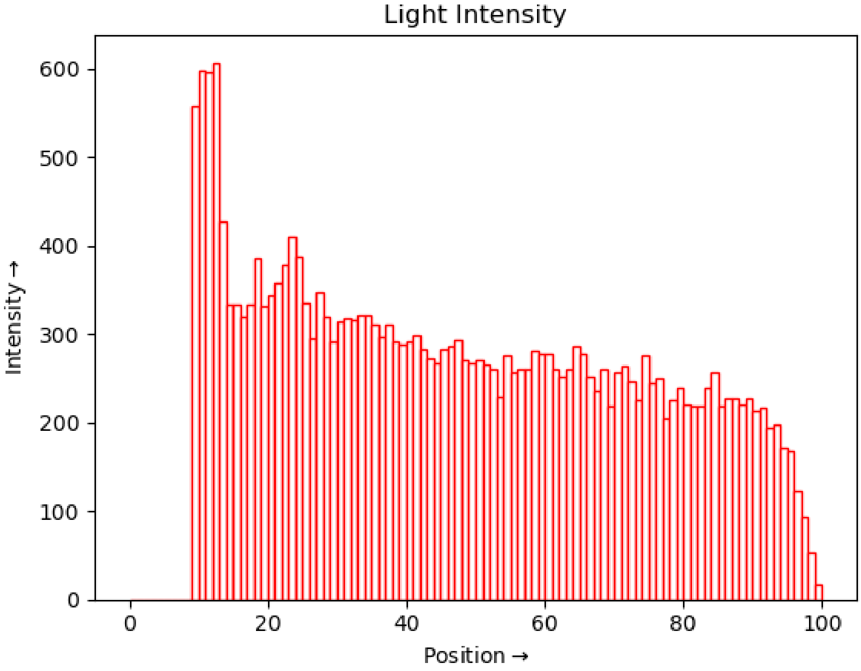
In this part we plot the electron density, intensity of light emitted and the electron phase space plots. We use the hist function to plot histograms for electron density and intensity.

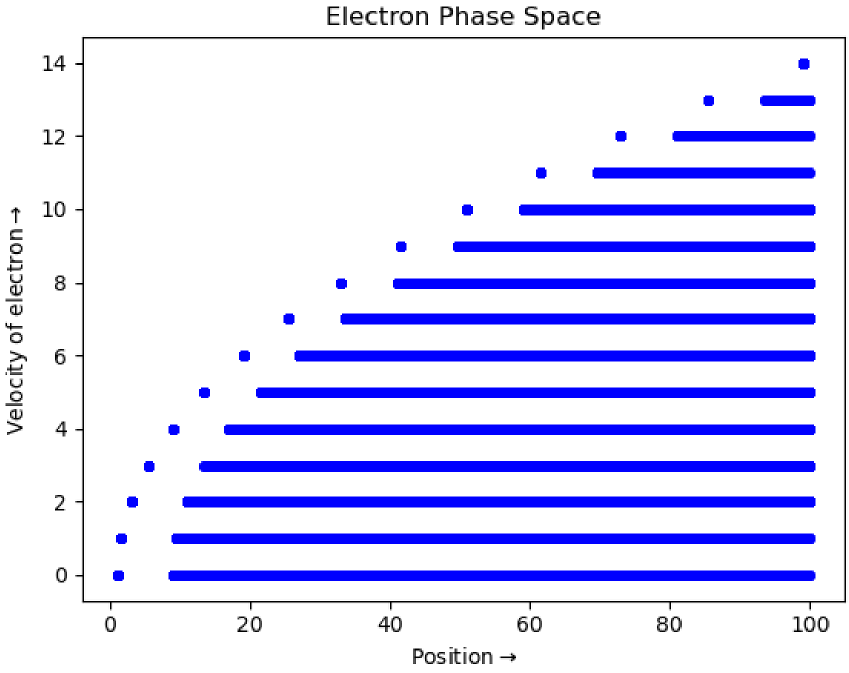
```
# Plotting the Electron Density Histogram
figure(0)
hist(X, bins = arange(0, 101, 1), rwidth = 0.8, color = 'g')
title('Electron Density')
xlabel('Position$\\rightarrow$')
ylabel('Number of electrons$\\rightarrow$')
show()

# Plotting the Light Intensity Histogram
figure(1)
pop_counts,bins,_ = hist(I, bins = arange(0, 101, 1), rwidth = 1.5, color = 'white',
↪   ec = 'red')
title('Light Intensity')
xlabel('Position$\\rightarrow$')
ylabel('Intensity$\\rightarrow$')
show()

# Plotting the Electron Phase Space
figure(2)
plot(X, V, 'bo', markersize = 4)
title("Electron Phase Space")
xlabel('Position$\\rightarrow$')
ylabel('Velocity of electron$\\rightarrow$')
show()
```


Electron Density
Number of electrons →
0
5000
10000
15000
20000
0
20
40
60
80
100
Position →


Light Intensity
Intensity →
0
100
200
300
400
500
600
0
20
40
60
80
100
Position →



2.4 Intensity Table:

To print the intensity table we make use of *pop_counts* and *bins* returned by the hist function for intensity plot. *bins* contains the dividing position between bins, so we convert it to middle point values.

```
# finding the middle point of the bins
xpos = 0.5*(bins[0:-1] + bins[1:])

# Printing the Intensity table
print("Intensity data:\nxpos\tcount")
for i in range(len(bins) - 1):
    print(" ", xpos[i],"\t ",int(pop_counts[i]),sep="")
```

Intensity Data	
xpos	count
0.5	0
1.5	0
2.5	0
3.5	0
4.5	0
5.5	0
6.5	0
7.5	0
8.5	0
9.5	558
10.5	598
11.5	596
12.5	607
13.5	427
14.5	333
15.5	334
16.5	320
17.5	333
18.5	386
.	.
.	.
.	.
88.5	221
89.5	228
90.5	214
91.5	217
92.5	195
93.5	197
94.5	171
95.5	168
96.5	123
97.5	94
98.5	54
99.5	16

3 Conclusion

From the intensity plot we can see that there are no collisions in the initial region of the tube light.The region upto 10 is where electrons are building up their energy. Beyond that is a region where the emission decays, representing the fewer energetic electrons that reached there before colliding. There are other peaks too. But these are diffuse peaks since the zero energy location of different electrons is different.