EE2703: Applied Programming Lab

Week 6: Simulations

Author: Surya Prasad S, EE19B121

April 11, 2021

1 Abstract:

In this week's assignment, we shall be simulating a tube light in a one dimensional model. We shall plot the following:

- The light intensity as a function of position at steady state and identify "dark spaces".
- Electron density as a function of position
- Phase Space diagram of the electrons

We shall make the following assumptions for the simulation:

- The electron comes to rest collision.
- The acceleration due to the electric field is $1m/s^2$.
- The time between each update of displacement and velocity is 1 sec.
- The collision point distribution of colliding electrons is assumed to be uniform.

2 Procedure:

First, we shall create a simulation universe. The tube is divided into n sections. At each instant of time, around M electrons are injected. We run the simulation for nk turns and check for electronic collisions which result in a photon emission. The following are the default parameters whose values can be changed in tbe commandline. The commandline accepts a maximum of 6 arguments.

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

We shall define variables to hold information related to the simulation.

```
# Variables to hold electron information
xx = zeros(n * M)          # Electron position
u = zeros(n * M)           # Electron velocity
dx = zeros(n * M)          # Displacement in current turn


# Variables to hold information for simulation
## In each turn, we record this information and since we don't know its length
    we are storing them as lists and extending them as required.
I = []                     # Intensity of emitted light
X = []                     # Electron position
V = []                     # Electron velocity
```

Now, we shall run the simulation for nk turns. In each loop we perform the following operations:

1. For the electrons whose position is greater than zero, we calculate the change in position and speed taking the acceleration to be $1m/s^2$ and Δt as $1s$. Numpy's where function can be used to identify the electrons. Code snippet to perform this operation:

```
ii = where(xx > 0)
dx[ii] = u[ii] + 0.5
xx[ii] += dx[ii]
u[ii] += 1
```

2. Determine the particles which have hit the anode (their positions would be beyond n). Again use the where command for this. The positions, displacements and velocities of these particles become zero. Code snippet to perform this operation:

```
reached_end = where(xx[ii] > n)[0]
xx[ii[reached_end]] = 0
u[ii[reached_end]] = 0
dx[ii[reached_end]] = 0
```

3. For the electrons whose velocity is greater than or equal to the threshold, we identify the electrons which are ionized. Their velocities become zero and their positions are added to I vector. The actual point of collision may have occured at any point between its current position and previous position and so we take a random number between those two points. Code snippet to perform this operation:

```
## Checking if any of the electrons achieved more than threshold speed
### For these electrons some of them collide based on probability p and
    their velocties become 0
kk = where(u >= u0)[0]
ll = where(rand(len(kk)) <= p)[0]
kl = kk[ll]
u[kl] = 0

### Determining colision points
rho = rand(len(kl))
xx[kl] -= (dx[kl] * rho)

### Adding the position of the excited atoms to I vector
I.extend(xx[kl].tolist())
```

4. Now, inject M new electrons. First we shall determine the actual number of electrons injected as

```
m = int(randn() * Msig + M)
```

   For these electrons, their positions will be initialised to 1 and their velocities will be zero. Code snippet to perform this operation:

```
not_ii = where(xx == 0)                    # Indices where position is 0
free_slots = min(len(not_ii), m)           # Number of free slots in which electrons got i
xx[not_ii[:free_slots]] = 1
u[not_ii[:free_slots]] = 0
dx[not_ii[:free_slots]] = 0
```

5. All the information of the position of the electrons and their velocities are now added to X and V vectors. Code snippet to perform this:

```
X.extend(xx.tolist())
V.extend(u.tolist())
```

6. Numpy's where function is a complicated code and we shall try to optimise its usage in the for loop using the following code snippet:

```
ii = array(list(all_ii - set(not_ii[0])), dtype=int)
```

where *all_ii* is the set of all position indices of the electrons.

3 Plot analysis:

Let's first plot the electron density as a histogram.

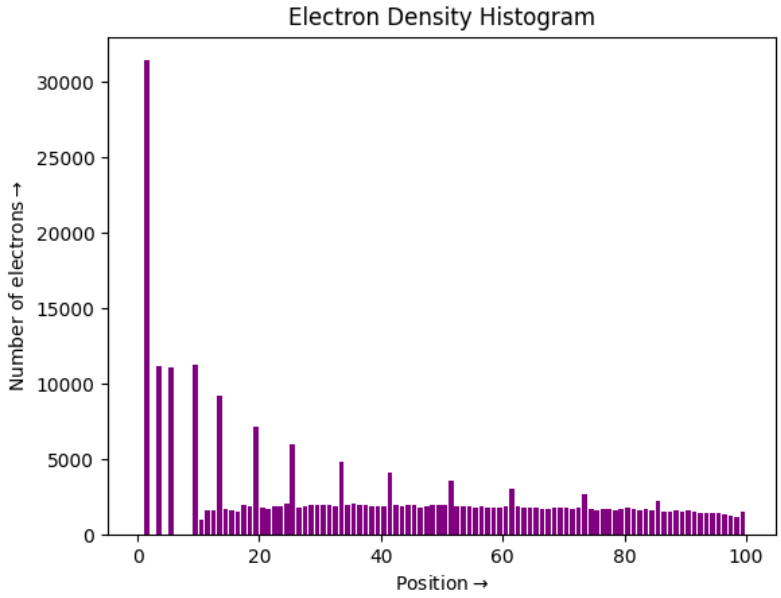

Figure 1: Electron Density Histogram

Now we shall plot a population plot of the light intensity using a histogram.

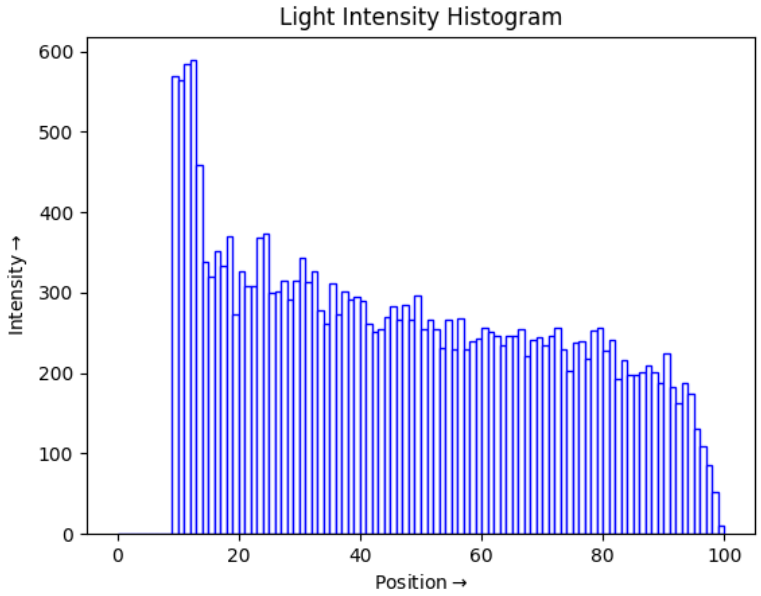

Figure 2: Light Intensity Histogram

Finally, we shall plot the electron phase space.

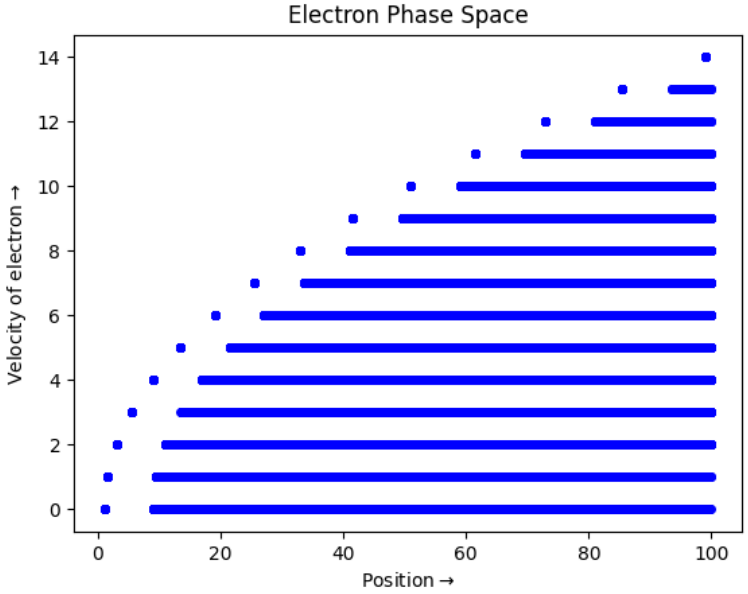

Figure 3: Electron Phase Space

4 Result:

Let's have a look at the Light Intensity got from the histogram. Rest of the data can be got in the code.

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	570
10.5	565
11.5	585
12.5	589
13.5	459
14.5	329
15.5	320
16.5	351
17.5	333
...	...
98.5	53
99.5	11

5 Increasing accuracy of simulation:

For the above plots we have assumed collision points to be uniformly distributed but the model is more accurate if we had assumed the time of collision to be uniformly distributed. So the update equation would be:

$$dx = udt + \tfrac{1}{2}(dt)^2 + \tfrac{1}{2}(1 - dt)^2$$

Here dt is uniformly distributed in time. After dt time, the electron accelerate after collision.

6 Conclusion:

So we have successfully simulated a 1-D model of a tube light. Light Intensity, Electron density and the Electron Phase space diagrams were plotted. In the regions where the electron has not reached threshold velocity there will be dark spots. This is given by:

$$x = \frac{1}{2}\frac{u_0^2}{a}$$

In our model, a (acceleration) is 1. So a dark spot will exist upto $x = \frac{1}{2}u_0^2$.

The Electron Phase Space diagram can be seen to bounded by a parabolic curve as expected.