Python Assignment -4

Swapnil Basak

EE11B122, IIT Madras

Abstract

We use a 1-Dimensional model of the tubelight. 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. We will conclude with population plots and the intensity data table.

Program

We include the shebang and import out libraries

Code

1a ⟨*code* 1a⟩≡

```
#! usr/bin/python

import sys
from scipy import *
from matplotlib.pyplot import *
from matplotlib.mlab import *
```

Now, we facilitate taking arguements from the command line and account for erraneous input. We also declare our vector arrays with all zeroes.

1b ⟨*declaration* 1b⟩≡

```
# Take arguements from command line
try:
    n=int(sys.argv[1]) # spatial grid size
    M=int(sys.argv[2]) # number of electrons injected per turn
    nk=int(sys.argv[3]) # number of turns to simulate
    u0=int(sys.argv[4]) # threshold activity
    p=float(sys.argv[5]) # probability that ionisation will occur
except:
    print "Invalid values"
    n=100
    M=5
    nk=500
    u0=7
    p=0.5
    print "Taking default values n=%d, M=%d, nk=%d, u0=%d, p=%f" %(n,M,nk,u0,p)

xx=zeros(n*M) # Electron Position
u=zeros(n*M) # Electron Velocity
dx=zeros(n*M) # Displacement in current turn
I=[] # Intensity
X=[] # Electron position
V=[] # Electron velocity
```

This is our for loop. We carry out several simultaneous processes here.

1. We first find all electrons which are in the chamber. We add their positions and velocities to the respective arrays.
2. We check for electrons that have reached the end of the chamber and reset them.
3. We then check for electrons that have more than critical velocity and are capable of ionizing the atoms
4. We use an approximate method to find out how many such electrons will actually ionize.
5. We then update all electrons in the chamber to our arrays.

2 ⟨*loop* 2⟩≡

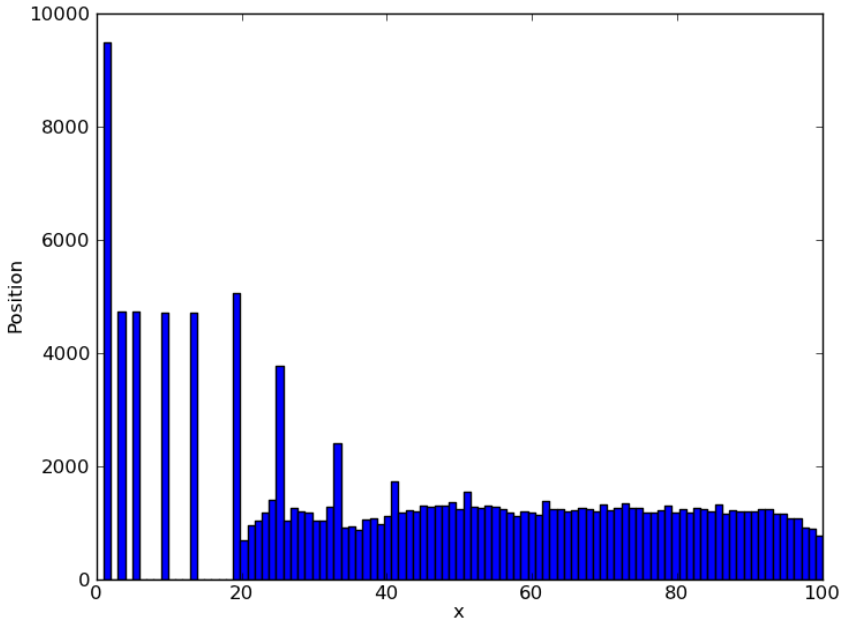
```
for k in range(nk):

        # Check for electrons which are in the chamber
        ii=where(xx>0);
        dx[ii]=u[ii]+0.5
        xx[ii]=xx[ii]+dx[ii]
        u[ii]=u[ii]+1

        # Check for electrons that have reached the end
        end=where(xx>=n)
        xx[end]=0
        u[end]=0
        dx[end]=0

        # Check for electrons having more than critical velocity
        active=where(u>=u0)
        ion=where(rand(len(active[0]))<=p);
        index=active[0][ion]
        u[index]=0
        xx[index]=xx[index]-rand(len(index))*dx[index]
        I.extend(xx[index].tolist())

        # Random distribution of injected electrons
        pos=where(xx==0)
        if len(pos[0])<M:
        xx[pos]=1
        else:
        xx[pos[0][:randn()*2+10]]=1
        pos = where(xx>0)

        # Position and velocity
        X.extend(xx[pos].tolist())
        V.extend(u[pos].tolist())
```

These are the plotting functions that plots the following:

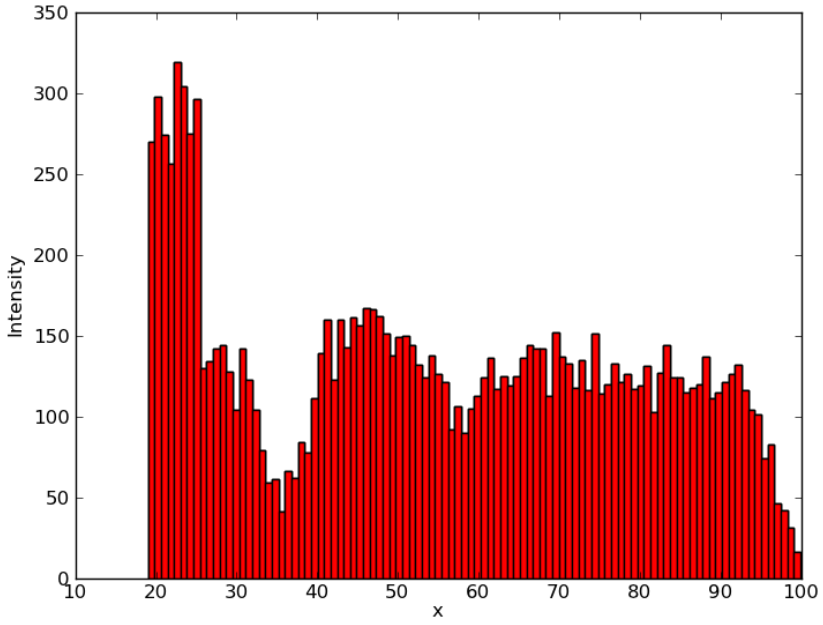
- Population plot of electron position.
- Population plot of electron intensity.
- Plot of electron's position and velocity
- Plot off the array of the history of all positions and velocity.
- We also form a two column matrix of intensity and electron count and store it in a file.

3 ⟨*plot* 3⟩≡

```
# Population plot
figure(0)
hist(X,bins=n)

# Intensity plot
figure(1)
hist(I,bins=n)
xlabel(r"x")
ylabel(r"I")

# Obtaining table from hist and saving
bins=hist(I,bins=n)[1]
xpos=(0.5)*(bins[0:-1]+bins[1:])
count=hist(I,bins=n)[0]
x=hstack((xpos.reshape((n,1)),count.reshape((n,1))))
savetxt("intensity.dat",c_[xpos,count])

# Plotting electron phase space
figure(2)
plot(xx,u,"x")
xlabel('x')
ylabel('velocity')

# Plotting electron phase space (TOTAL)
figure(3)
plot(X,V,"x")
xlabel('x')
ylabel('velocity')

show()
```

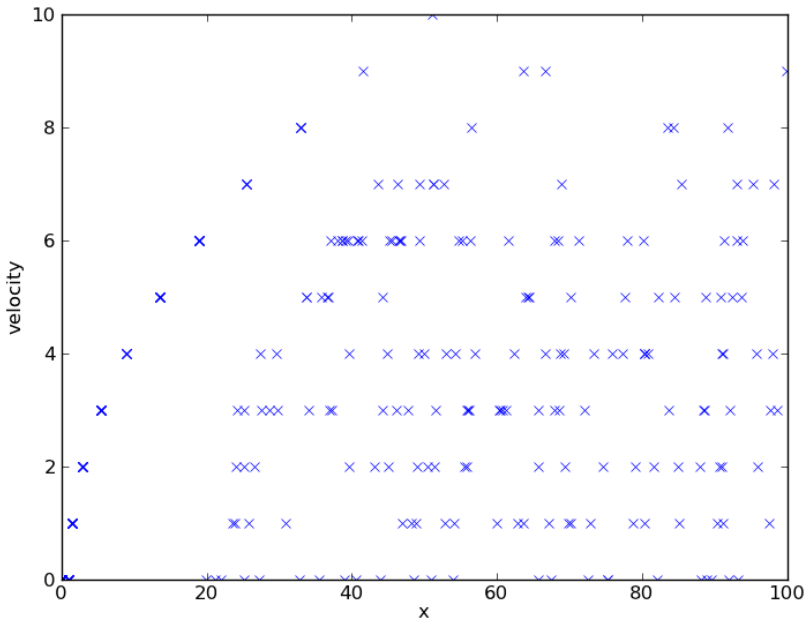


Position plot

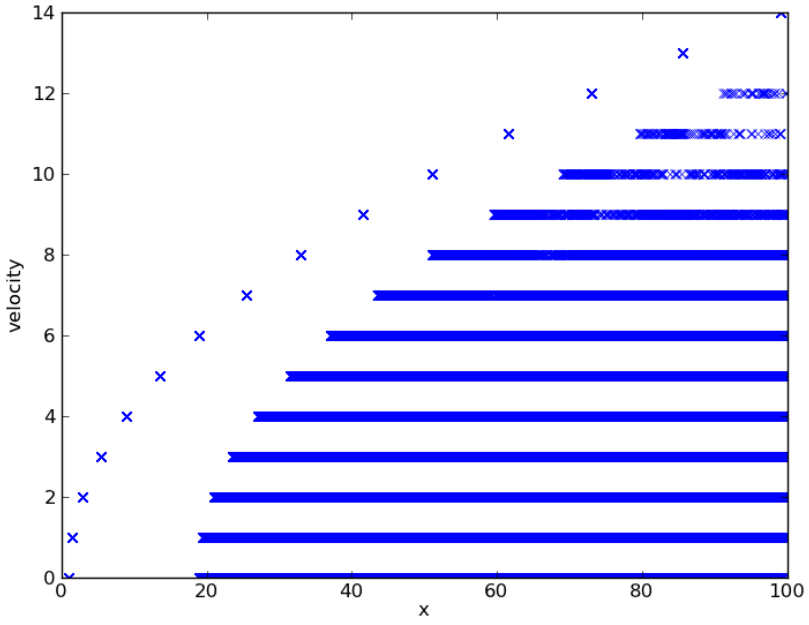


Intensity Plot

Figure 1: Position and Intensity Plots



After simulation



For all time

Figure 2: Electron Phase plots