Nanonetworking Homework

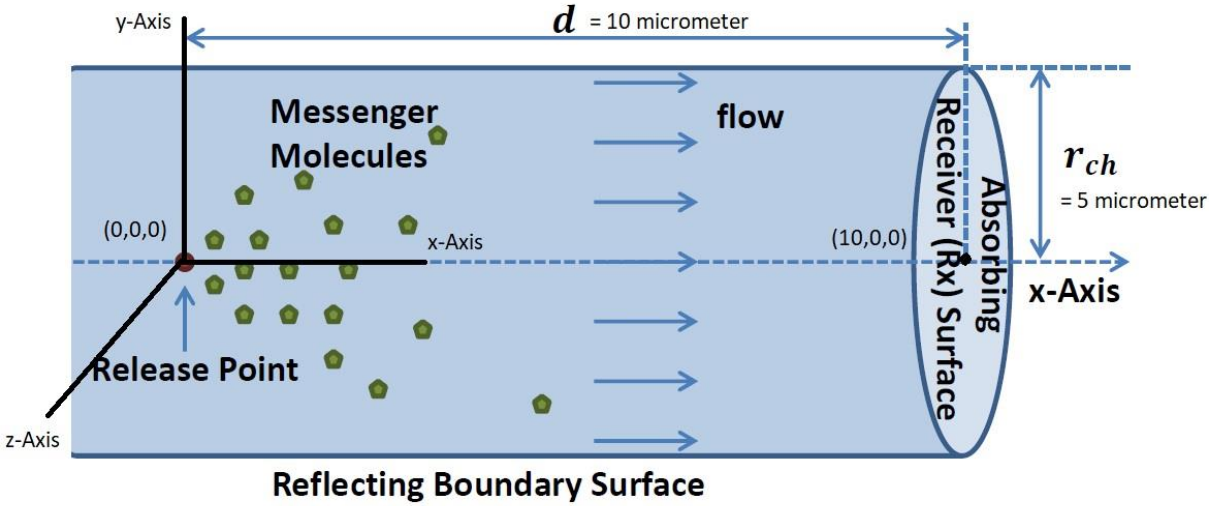
Halil Umut Özdemir 2016400168

Problem:

In this project, problem is simulating a cylindrical vessel-like environment with a positive constant flow towards the receiver. The diffusion model consists of a point transmitter, a fully absorbing circular receiver, a single type of information carrying messanger molecules, and a vessel-like environment with uniform flow. The vessel-like environment is considered to be a perfect cylinder with a complete reflecting surface.

At the end, I am expected to analyze the effect of flow over intersymbol interference (ISI) by changing flow velocity.

Designing the Simulation:



This simulation, I take the release point as the origin of the coordinate system. The distance between releasing point and absorbing surface is 10 micrometer and radius of the channel is 5 micrometer as expected. The motion of a molucule is simulated as a Gaussian random variable with 0 mean and 2 * Diffision Coefficient(79.4 μm2/s) variance. Flow is a 3 dimensional vector because it has only x component, y and z components of flow are 0. In this simulation, the reflecting surface type is assumed as rollback.

Implement the Code:

The code of simulation has 3 parts: Initialize the simulation variables, Simulation loop, Interpreting the results as plot.

1. Initialize the simulation vaiables:

The number of molecules in this simulation is 10 000. Distance between release point and reciever is 10 micrometers. Radius of the cylinder channel is 5 micrometers. The receiver cordinates is represented as a 1x3 matrix [10,0,0]. Diffusion coefficient is 79.4 μm2/s. Because this simulation is made with iterating time steps, size of a time step is initialized as 0.0001 seconds. Total time is initialized as 2 seconds. The flow is 3 dimensional vector. The mol_matrix represents the positions of molecules at any time step during the simulation. Arrival times stores the absorbing time of a molecule if absorbed, if not it stores 0 for this molecule.

```
%Number of molecules.
num_of_mols = 10000;
%Distance between release point and reciever(Micrometers).
distance = 10;
%Radius of cylinder channel(Micrometers).
channel_radius = 5;
%Coordinates of the cicular reciever(Micrometers).
reciever_coordinates = [distance, 0, 0];
%Diffusion coefficient(Micrometers^2/seconds).
coef = 79.4;
%Step size. (Seconds).
step = 10^-4;
%Total time. (Seconds).
time = 2;
%In micrometers per second
flow = [0, 0, 0];
%Molecule x, y, z coordinate matrix
mol_matrix = zeros(num_of_mols, 3);
%Arrival times of the molecules.
arrival_times = zeros(num_of_mols,1);
%Standart deviation of the gaussian diffusion motion.
std = sqrt(2 * coef * step);
```

2. Simulation Loop:

This loop iterates over all time steps in total time. Loop starts with genetaing displacement vectors for all molecules in simulation. It generates this vectors with using the Gaussian random variable with 0 mean and 2 * Diffision Coefficient(79.4 μm2/s) variance. After that, mol_matrix is copied, because of the rollback assumption. Then the displacement and flow vectors are added to the position matrix of molecules(mol_matrix). For calculating a molecule is reflected or not, y and z columns of mol_matrix is copied. The reflected molecules are calculated; if sum of squares of y and z coordinates is bigger than the square of channel raduis, then this molecule is a reftected molecule. All reflected molecules are taken back to their previous position before the current time step. Then the absorbed molecules are calculated with respect to their x coordinate. If x coordinate is bigger than the distance between receiver and releasing point, then this molecule is absorbed. At the end of each iteration, arrival time of all absorbed molecules are stored in arrival_time matrix.

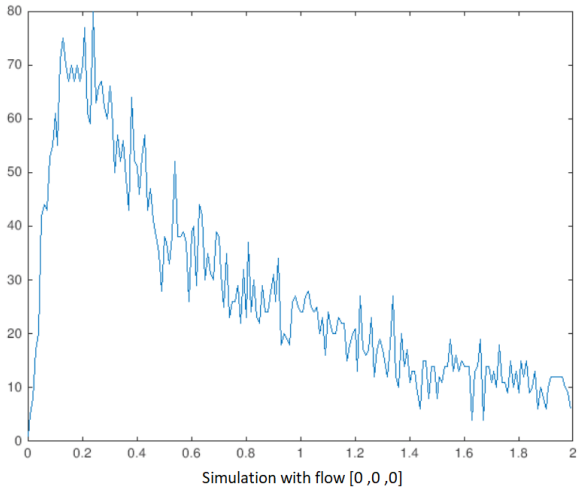
```
%Simulation Loop
for i = 1:time/step

    %Generates  movements of current time step.
    mov_vectors = normrnd(0,std, size(mol_matrix));

    %Takes a copy of molecule matrix before calculations. Because if a
    %particle bumps it bounce back to its previous place.
    prev_copy_mol_matrix = mol_matrix;

    %Calculates the new position of molecules by adding random movements
    %and current flow.
    mol_matrix = mol_matrix + mov_vectors + (flow * step);

    %Takes the y and z cordinates of all molecules for calculating
    %molecules that bounces its previous position.
    mol_matrix_yz = mol_matrix(:,2:3);

    %Squares y and z cordinates for all particles.
    mol_matrix_yz = mol_matrix_yz.^2;

    %Calculates the sum of squares for all rows(particles).
    sum_vector = sum(mol_matrix_yz,2);

    %Finds the particles which exceeds the channel. If sum of squares of y
    %and z cordinates is bigger than channel radius, then it is a exceeding
    %particle.
    %Exceeding Particles is a logical vector that holds 0 for no-exceeds, 1 for exceeds.
    exceeding_particles = find((arrival_times==0) & (sum_vector>channel_radius^2));

    %Bounce back to their previous position.
    mol_matrix(exceeding_particles,:) = prev_copy_mol_matrix(exceeding_particles,:);

    %Finds the number of molecules absorbed by receiver.
    %If the x cordinate is bigger than or equal to the distance between
    %releasing point an receiver, then it is absorbed.
    %Absorbs is a logical vector that holds 0 for no-absorbs, 1 for absorbs.
    absorbs = find((arrival_times==0) &  (distance<=mol_matrix(:,1)));

    %Store the arrival times of particles(absorb vector has value 1 in that row).
    arrival_times(absorbs) = i*step;

end
```

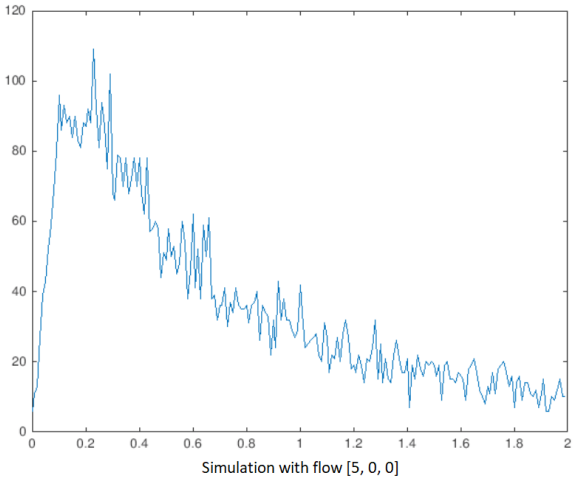
3. Interpreting The Results As Plot:

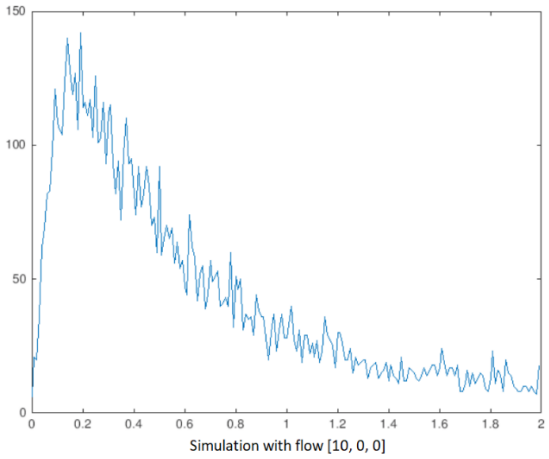
This part only plots the number of molecules absorbed by the time and the time step plot.

```
%Plotting the results.
%Drop initial zeros in arrival times array.
res = arrival_times(arrival_times ~= 0);


%Create X axis values and intervals for plotting.
xStart = 0;
dx = 0.01;
N = 200;
x = xStart + (0:N-1)*dx;

%Take histogram of values.
h = hist(res,N);
%Plot values
plot(x,h);
```

Analyze the Results:







Three plots above are the results of the simulation with flows 0, 5 and 10 micrometers per second. As we can see, all 3 plots takes the maximum value between 0.15 and 0.3 seconds. For all scenarios, from 0 to 0.3 seconds approximately number of molecules received increases. After the maximum value which is located between 0.2 and 0.4 seconds, the number of molecules received started to decrease. As we can see, for all 3 scenarios 2 seconds is not enough to run out the signal (the received molecules). For all 3 flow vectors, the shape of the plot is similar which means flow on x-axis doesn't change the shape of plot. But when the flow value increases, the number of received molecules increases too, which means when flow increases, plots shift upwards.