INDRAPRASTHA INSTITUTE of INFORMATION TECHNOLOGY DELHI

HOMEWORK ASSIGNMENT - 6

INDRAPRASTHA INSITUTE OF INFORMATION TECHNOLOGY, DELHI

COMPUTER SCIENCE AND APPLIED MATHEMATICS

Introduction to Quantitative Biology (BIO213)

Author:
Aditya Chetan
(2016217)
Pranav Jain
(2016255)
Brihi Joshi
(2016142)

Date: March 15, 2018

1 Solution

In our solution we have displayed the following on the console:

- Number of molecules: The number of molecules participating in the random walk.
- Size of the lattice: The size of the lattice in which the molecules are placed.
- Number of iterations: The number of iterations for which we perform the walk.
- Mean X - displacement: The mean displacement along the X - direction.
- Mean Y - displacement: The mean displacement along the Y - direction.
- Mean displacement vector: A vector denoting the mean displacement.
- Magnitude of mean displacement ($< r >$): Magnitude of the mean displacement vector.
- Mean square displacement ($< r^2 >$): Magnitude of the mean square displacement.

As can be seen in the output attached, the magnitude of the mean displacement, $< r >$ should be close to 0 and in our case, it is 0.8302, which is approximately 0. Hence, the simulation was successful.

2 Challenge Problem Solution

For the challenge problem, 3 probability distribution graphs have been plot for 250, 500 and 750 iterations respectively.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% BIO213 - IQB
%
% Homework Assignment 6
%
% Brihi Joshi (2016142)
% Aditya Chetan (2016217)
% Pranav Jain (2016255)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


analysis(100, 1000, 50);




%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
%This function initialises a list of x-coordinates and y-coordinates of
% molecules in the lattice.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [x_coords, y_coords] = initialize_coords(n, size)

    x_coords = zeros(1, n);
    y_coords = zeros(1, n);
    i = 1;

    while i < (n + 1)
        x = randi(size, 1);
        y = randi(size, 1);
        flag = 0;
        for j = 1 : i-1
            if x_coords(j) == x && y_coords(j) == y
                flag = 1;
                break;
            end
        end

        if flag == 1
            continue;
        else
            x_coords(i) = x;
            y_coords(i) = y;
            i = i + 1;
        end
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% The function moves molecules left, right, up or down, depending upon the
% direction code provided. The direction codes are as follows:
%
% 0 -> Up
% 1 -> Right
% 2 -> Down
% 3 -> Left
%
% In case the destination in a particular direction is already occupied by
% another molecule or the molecule is at the boundary of the lattice, then
% it is not moved.
```

```
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [x, y] = update_coords(xi, yi, dir, x_coords, ...
    y_coords, size, index)

    x = xi;
    y = yi;
    if dir == 0
        if yi == size
            return;
        else
            for i = 1 : length(x_coords)
               if (x_coords(i) == xi && y_coords(i) == yi + 1)
                   return;
               end
            end
            x = xi;
            y = yi + 1;
            y_coords(index) = y;
        end

    elseif dir == 1
         if xi == size
            return;
        else
            for i = 1 : length(y_coords)
               if (x_coords(i) == xi + 1 && y_coords(i) == yi)
                   return;
               end
            end
            x = xi + 1;
            y = yi;
            x_coords(index) = x;
         end
    elseif dir == 2
        if yi == 1
            return;
        else
            for i = 1 : length(y_coords)
               if (x_coords(i) == xi && y_coords(i) == yi-1)
                   return;
               end
            end
            x = xi;
            y = yi - 1;
            y_coords(index) = y;
         end

    elseif dir == 3
        if xi == 1
            return;
        else
            for i = 1 : length(y_coords)
               if (x_coords(i) == xi - 1 && y_coords(i) == yi)
                   return;
               end
            end
            x = xi - 1;
            y = yi;
            x_coords(index) = x;
         end

    end
```

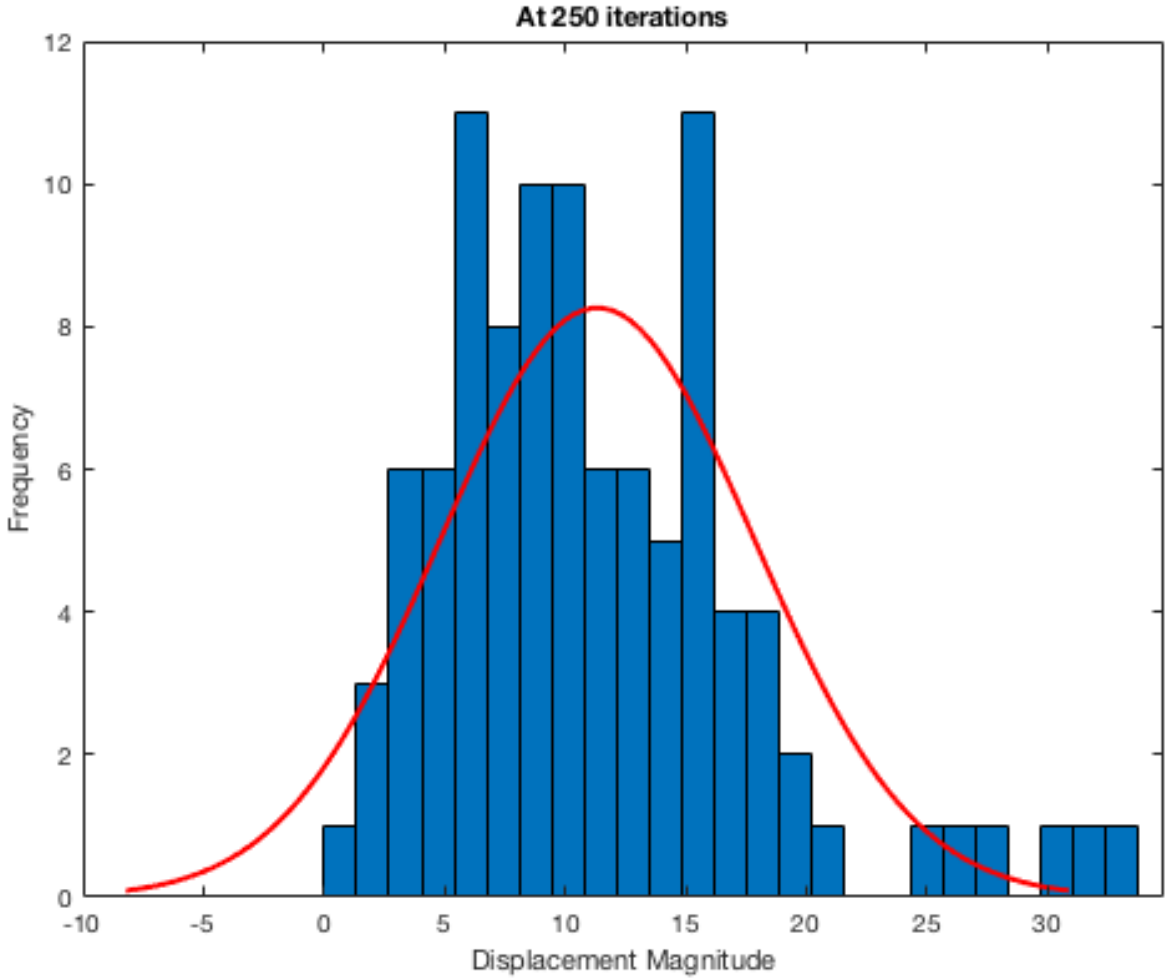
```
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% This function simulates a random walk from the number of molecules,
% number of iterations and the size of the lattice
%
% Here, we are also plotting the plots of the magnitude of the mean
% displacement at 3 different points too, i.e., after 250, 500, and 750
% iterations respectively. Here, we see that the plots approximate to a
% Gaussian distribution in accordance with CLT.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [copy_x, copy_y, xs, ys] = random_walk(n, iter, size)

    [x_coords, y_coords] = initialize_coords(n, size);
    copy_x = x_coords;
    copy_y = y_coords;
    figure;
    for i = 1 : iter
        for p = 1 : n
            dir = 0;
            chance = rand();
            if chance < 0.25
                dir = 0;
            elseif chance < 0.50
                dir = 1;
            elseif chance < 0.75
                dir = 2;
            elseif chance < 1.00
                dir = 3;
            end

            [a, b] = update_coords(x_coords(p), y_coords(p), dir, ...
                x_coords, y_coords, size, p);
            x_coords(p) = a;
            y_coords(p) = b;


        end

        if i == 250
                rx = x_coords - copy_x;
                ry = y_coords - copy_y;
                r = sqrt(rx.^2 + ry.^2);
                figure(1);
                histfit(r,25, 'normal');
                xlabel('Displacement Magnitude');
                ylabel('Frequency');
                title('At 250 iterations');

            elseif i == 500
                rx = x_coords - copy_x;
                ry = y_coords - copy_y;
                r = sqrt(rx.^2 + ry.^2);
                figure(2);
                histfit(r,25, 'normal');
                xlabel('Displacement Magnitude');
                ylabel('Frequency');
                title('At 500 iterations');


            elseif i == 750
                rx = x_coords - copy_x;
```

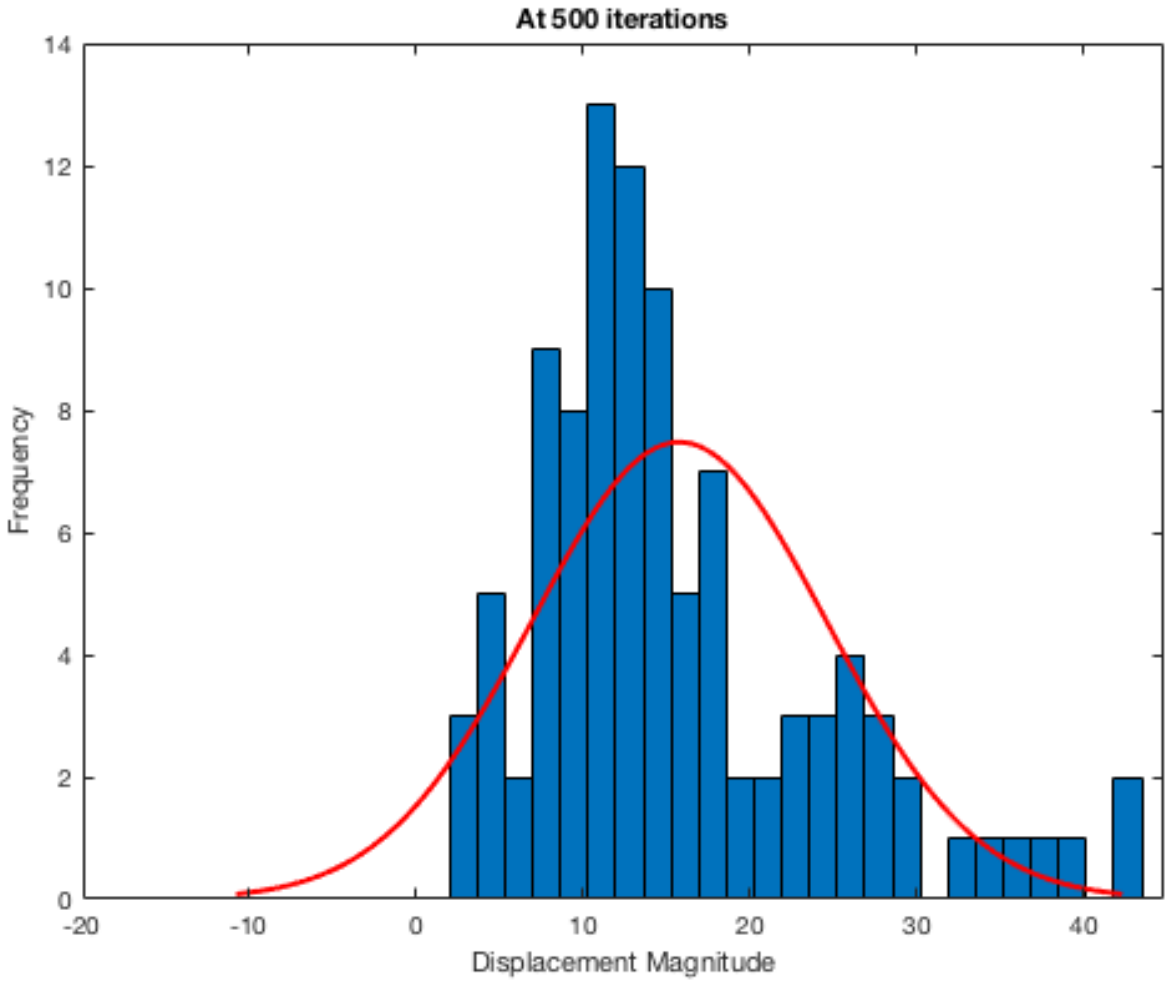
```
                ry = y_coords - copy_y;
                r = sqrt(rx.^2 + ry.^2);
                figure(3);
                histfit(r,25, 'normal');
                xlabel('Displacement Magnitude');
                ylabel('Frequency');
                title('At 750 iterations');

        end
    end

    xs = x_coords;
    ys = y_coords;

end

function [rx, ry, r2] = analysis(n, iter, size)

    [xi, yi, xf, yf] = random_walk(n, iter, size);

    rx = mean(xf - xi);
    ry = mean(yf - yi);

    r = sqrt(rx^2 + ry^2);

    r2 = mean((xf-xi).^2)+mean((yf-yi).^2);

    fprintf("Following is the analysis of the random walk simulation:\n\n");
    fprintf("1. Number of molecules = %d\n", n);
    fprintf("2. Size of lattice = %d\n", size);
    fprintf("3. Number of iterations = %d\n", iter);
    fprintf("4. Mean X displacement = %.4f\n", rx);
    fprintf("5. Mean Y displacement = %.4f\n", ry);
    fprintf("6. Mean displacement vector = %.4fî + %.4fĵ\n", rx, ry);
    fprintf("7. Magnitude of mean displacement = %.4f\n", r);
    fprintf("8. Mean square displacement = %.4f\n\n", r2);



end
```

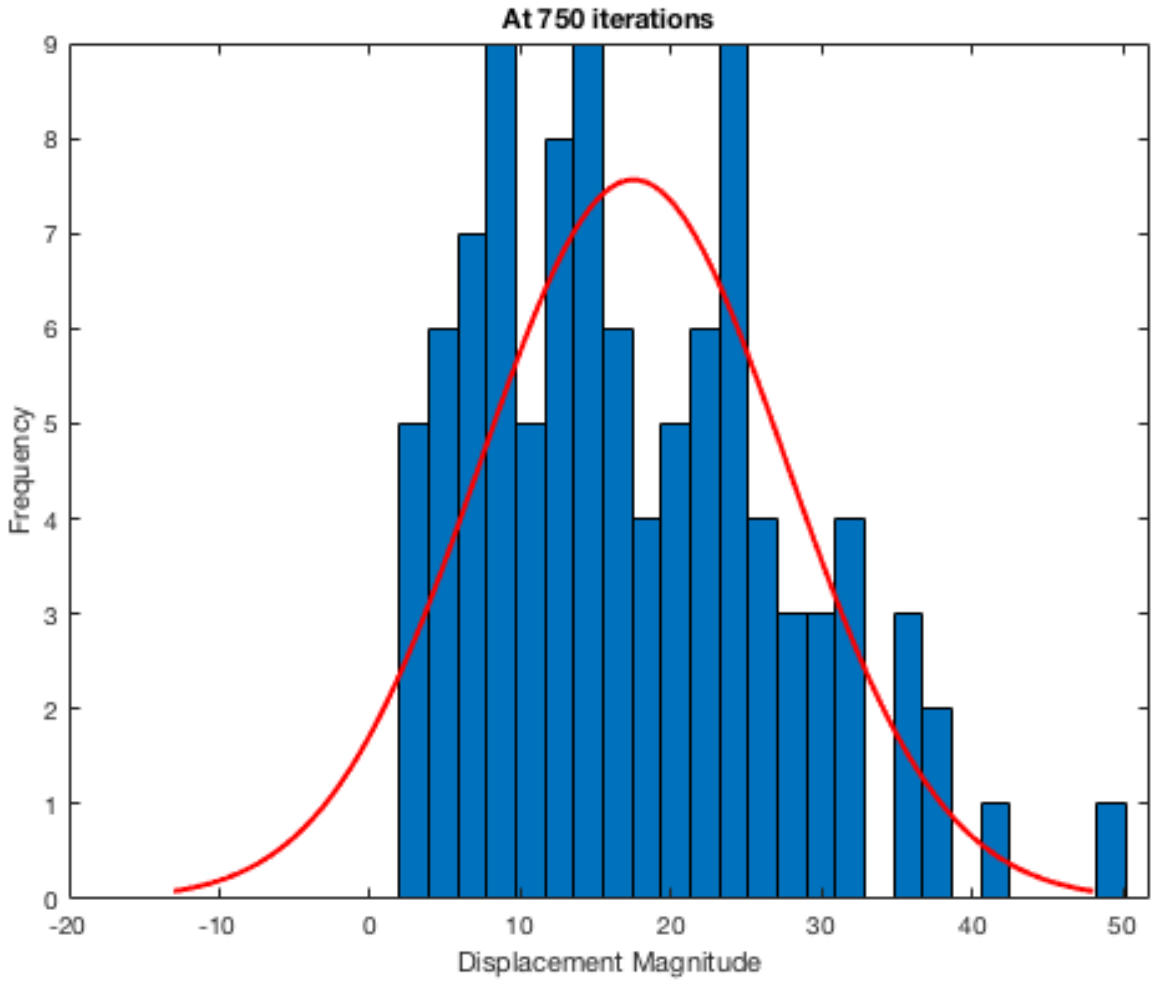
```
Following is the analysis of the random walk simulation:

1. Number of molecules = 100
2. Size of lattice = 50
3. Number of iterations = 1000
4. Mean X displacement = 0.0200
5. Mean Y displacement = -0.8300
6. Mean displacement vector = 0.0200î + -0.8300ĵ
7. Magnitude of mean displacement = 0.8302
8. Mean square displacement = 447.9100
```

□□□

Published with MATLAB® R2017b


At 250 iterations
Frequency
Displacement Magnitude
0
2
4
6
8
10
12
-10
-5
0
5
10
15
20
25
30


At 500 iterations
Frequency
Displacement Magnitude
0
2
4
6
8
10
12
14
-20
-10
0
10
20
30
40


At 750 iterations
Frequency
Displacement Magnitude
0
1
2
3
4
5
6
7
8
9
-20
-10
0
10
20
30
40
50