For the first part of the project, an algorithm for simulating a discrete random walk on a two dimensional grid was created. This was done by noting that depending on which position we are on the grid, the number of possible paths the point can take differs. Specifically, when the current position is on a corner, on an edge, or on neither, then the number of paths that can be taken are 3, 5, and 8 respectively.

When the current position is on a corner, the algorithm used to decide which path the point should take was described by a single arithmetic expression. Depending on whether the current position is on the edge of the x coordinate, y coordinate, or both, the expression properly adjusts the value to add to the current position by including a negative or positive symbol. For instance, in a 8x8 grid, if the current position is (0,0), we see that both the x and y coordinate are on the edge, so the only change that can be made to the current position is +1. If the current position is (0,7), we see that the only change that can be made to the current x position is +1, and the only change to the current y position is -1. In addition, the unidrnd() function was used to determine which of the 3 possible paths to take for each corner. The expression also takes this into account.

When the current position is an edge or neither a corner nor edge, the algorithm used to decide which path the point should take makes use of a lookup table. The lookup table consists of 8 possible x and y coordinate pair changes. For the case of an edge, another lookup table of 5 x and y coordinate pair changes was created, and the original lookup table was traversed to determine which x and y coordinate pair changes are relevant to the edge and should be added to the new lookup table. Depending on which edge the current position is on, the new lookup table will consist of different pairs of x and y coordinate changes. There are a total of 4 edge scenarios, the left, top, right, and bottom edges. The unidrnd() function was used to determine which of the 5 possible paths to take for each point on an edge. The value from the function serves as the index to the new lookup table, which will give a pair of values describing the change to be made to the current x and y coordinate values. For the case of neither a corner nor an edge, the unidrnd() function determines which of the 8 possible paths to take for each point. The function’s value serves as the index to the original lookup table, which gives a pair of values describing the change to be made to the current x and y coordinate values.

In simulating the discrete random walk, 10000 simulations were run in a 5x5 grid for 10000 time steps, where the starting point is (0,0). For each simulation, a variable was used to keep track of the number of time steps it took to reach the point (4,4). This yields a 1x10000 vector of values representing the number of time steps it took to reach the point (4,4). Plotting the histogram and the log normal distribution of this vector yields Figure 1. The mean, variance, and skewness of the data were found to be 13.2464, 81.5796, and 2.0982 respectively. The average number of times the position (2,4) was visited was found to be around 277.2576, which gives us its probability of being visited of 0.0277. This probability will not be the same for all states, because some states have a higher or lower chance of being entered. For instance, a corner can only be entered from 3 other points, whereas a point that is not a corner nor an edge can be entered from 8 other points. Clearly, the latter point has a greater chance of being entered.

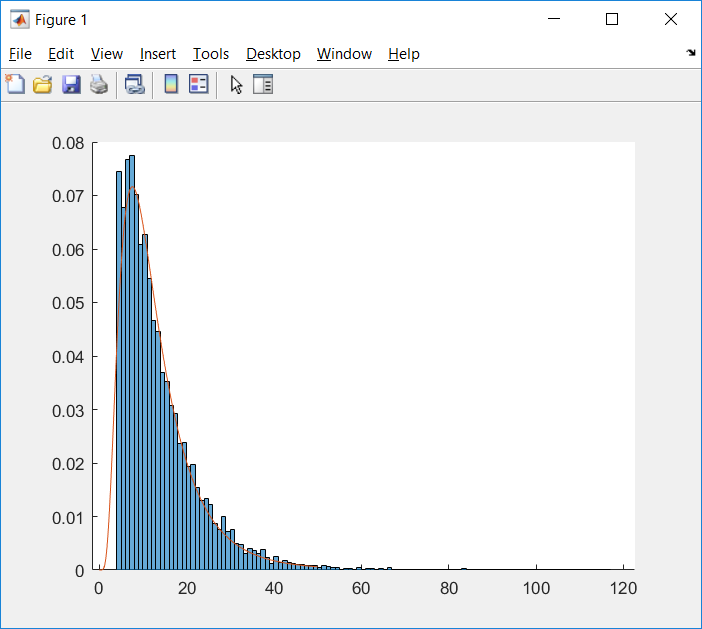


Figure 1 : Histogram and log normal distribution of the number of time steps to reach (4,4)