BML Functions

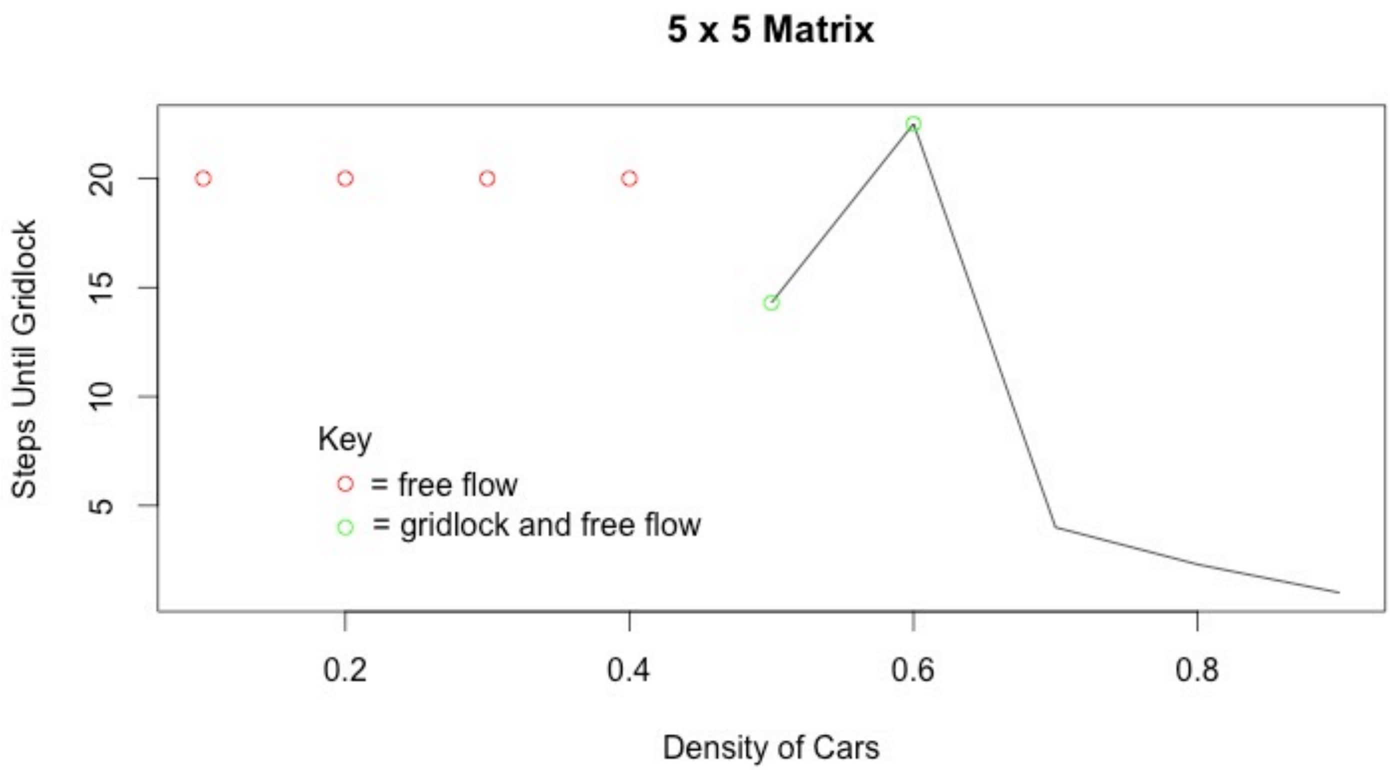
In order to create a visual system that moves red and blue blocks indefinitely or until gridlock is reached, I produced functions that would facilitate this process step by step. The first of these was `bml.init`, a function that creates a matrix. The values 0,1, and 2 are randomly placed in the matrix, with 1 and 2 appearing with the same probability. Thus, `bml.init` takes the input of number of rows and columns as well as a defined probability and returns the matrix. By using the function `image()` on the matrix and defining a palette of red, white, and blue, the matrix can be visualized as a grid where red cars are represented by 1, blue cars are represented by 2, and there is an empty white space at every 0.

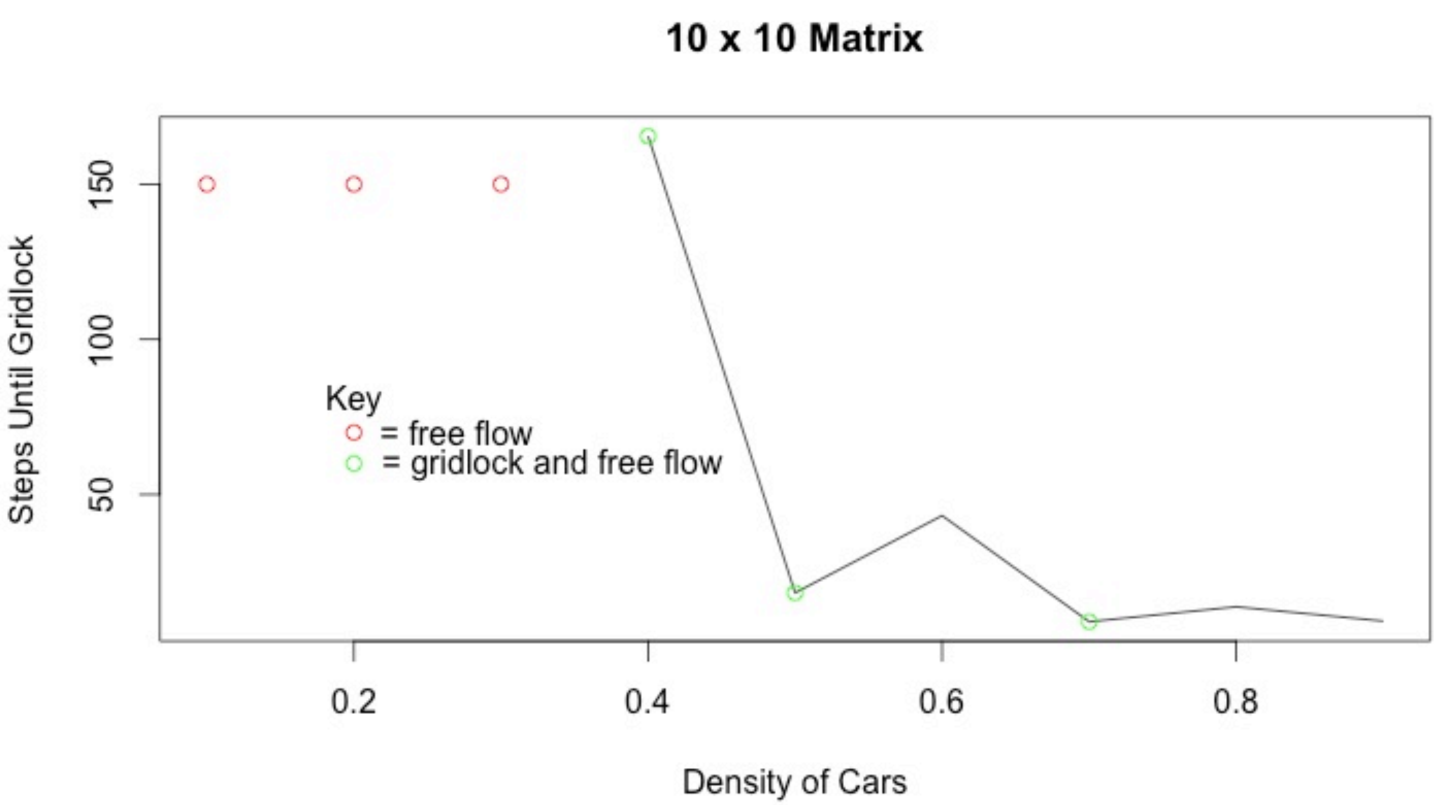
Next, I created a function `move.red`. For every 0 in the matrix with a 1 below it, the 1 will take its place. In addition, for every 0 in the bottom row with a 1 in the top row in the same column, the 1 would take its place in the bottom, as if recycling itself through the grid. I created a similar function `move.blue` that moves every 2 to the right if there is a 0 there. The next function, `two.step`, applies `move.blue` after `move.red`, so the blue blocks will move after the red blocks have. The function `bml.step` then applies `two.step` either indefinitely (free flow state) or until gridlock is reached. There is a counter within the function that lets the user know how many steps were taken until gridlock—if it was at all reached. If the grid is in free flow, then the function will simply not return a result.

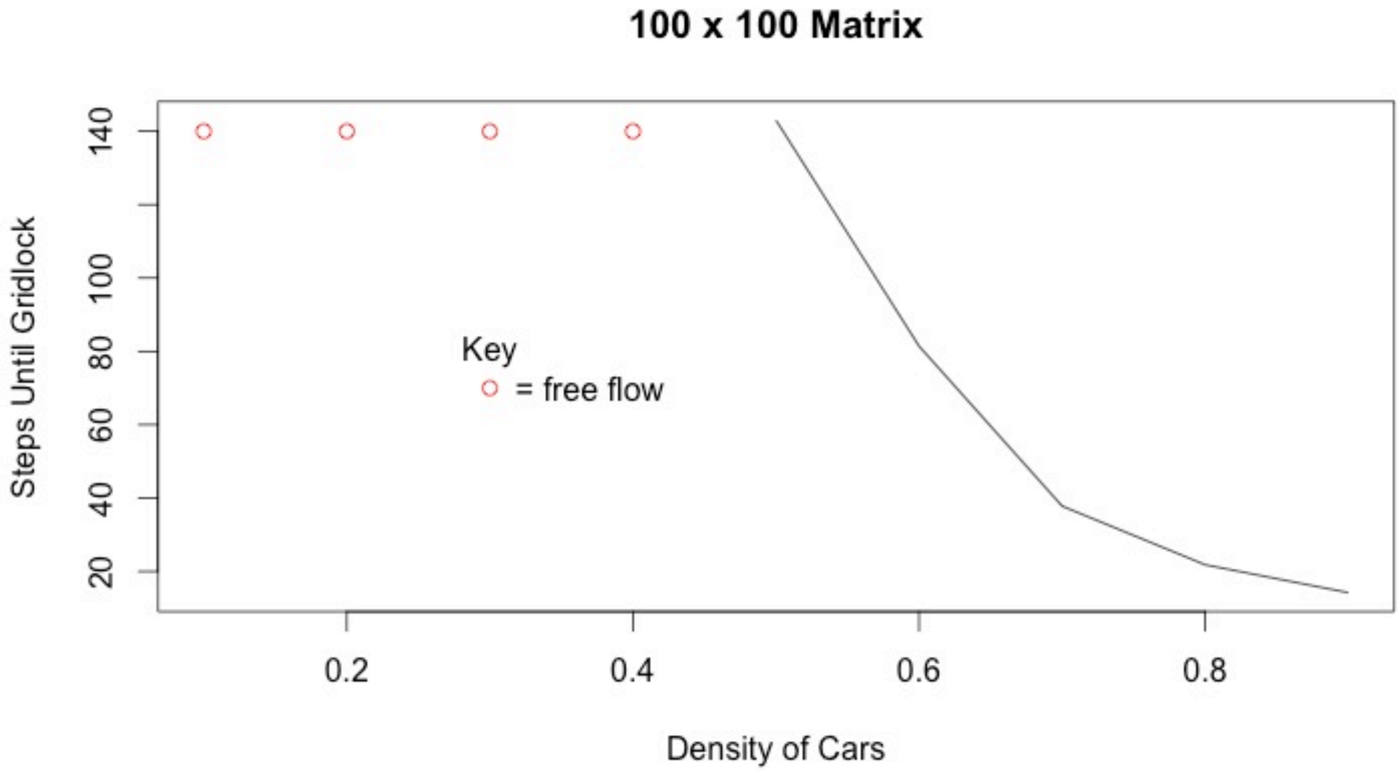
BML Simulations

After creating the functions necessary, I ran multiple simulations with various parameters to understand what affects whether the matrix will be in the free flow state or will reach gridlock. I experimented with three different sized matrices: 5x5, 10x10, and 100x100. Within each matrix, I ran five random matrices of each density (.1, .2, .3, .4, .5, .6, .7, .8, and .9). Lower densities usually resulted in free flow. Medium densities varied between free flow and gridlock—for example, three of the five samples might be free flow while the other two hit gridlock. Higher densities almost always resulted in gridlock.

BML Plots


5 x 5 Matrix
Steps Until Gridlock
5
10
15
20
Key
= free flow
= gridlock and free flow
0.2
0.4
0.6
0.8
Density of Cars


10 x 10 Matrix
Steps Until Gridlock
50
100
150
Key
= free flow
= gridlock and free flow
0.2
0.4
0.6
0.8
Density of Cars



*Note: In all the above graphs, the steps to reach gridlock are an average calculated from numerous trials.

BML Comments

There is clearly a negative correlation between density of cars and the number of steps is takes to hit gridlock. This makes logical sense because as the number of cars in a given area increases, there are less empty spaces for them to move into. This relationship is expressed in all three graphs.

What is most interesting about the data is the differences between the matrices. With the smallest matrix (5 x 5), unless the cars were in a free-flow state (such as at p=.1 through p=.4 and sometimes at p=.5 and p=.6), the average number of steps until gridlock was relatively small. Therefore, differences within each trial were more pronounced (for instance, it could take several steps to reach gridlock for one trial of p=.6 but very few steps for another trial of p=.6). Thus, there are fluctuations in the curve. This is also the case for the 10 x 10 matrix.

For the large matrix (100 x 100), the curve is significantly smoother. This is because the average number of steps until gridlock was much larger for this matrix, so differences within each set of trials for a particular density are more trivial.

In conclusion, a larger matrix helps the audience visualize the relationship between car density and average time until gridlock more efficiently.