PHY 200 Final Project

Mallory Justis

May 2022

1 Code

To use the code, run the main executable file. It should ask for input parameters. Enter the desired speed limit (mph), length of road being observed (miles), and how long the traffic light is green (minutes). The code will then ask for which speed-density relationship the user wants to observe, and they can enter one of the three options: Greenshield, Underwood, or Pipes-Munjal. If the Pipes-Munjal relationship is selected, the user will also be asked to input the value of n, which is the exponent of the density ratio in the formula. This allows the user to compare different n values of the Pipes-Munjal relationship if so desired.

The function will then produce a density plot, with time on the x axis and distance on the y axis.

Below these, the function should produce three graphs: a speed vs density plot, a speed vs flow plot, and a flow vs density plot.

2 The Process

Originally, in my project proposal, I had intended to get the code working for the first code submission and then expand upon that to make a two dimensional model. However, after extensive research, I realized that traffic engineering is much more complex than I thought, and creating a two dimensional model of an intersection was an unrealistic goal. So I decided to improve upon my first code submission by adding different speed-density models that could be compared. I also intended to add more lanes, but after further research, this would significantly change the code that I was working with.

The biggest issue with my original code submission was that when I tried to do a higher number of discretizations, for example, 100, the code would break down and there would be numbers outside of the range of Python's abilities. I additionally was getting negative values for density, and values that were much larger than 200 cars/mile (the maximum density). So, not to anyone's surprise,

my code was not quite right.

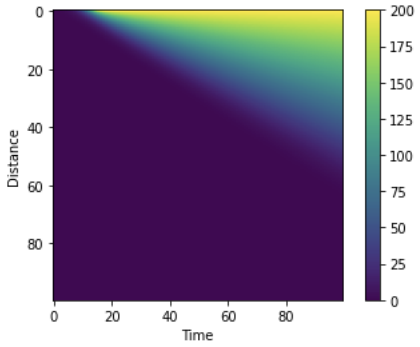
After extensive research of Traffic Engineering Master's Degree Theses (great reads, if I do say so myself), many late nights in the library, and six Redbulls later, I finally figured out the issues with my code. The first issue was that I had time in the inside of my nested loop, when it should be time on the outside and space on the inside (iterating over the positions in each time step, because they change with time). I was also updating the next position when I should have been updating the next time step when I used my numerical scheme. Updating this allowed for me to put in a higher level of discretizations. Changing the nested loop also fixed the issues I was facing with negative numbers.

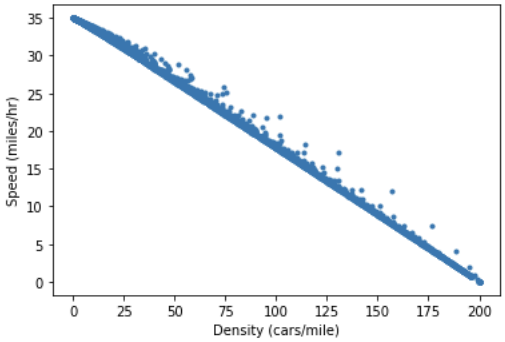
I then adjusted my speed-density relationships. The Greenshield relationship was working, but the Greenberg was still producing numbers outside of the range of Python, a lot of infinities. Because it is a logarithmic relationship, as density approaches 0, speed tends toward infinity. Because this model has a lot of situations in which the density is or is very close to zero, considering that we are looking at one way of traffic, this model was not a good fit for my code. I decided to add the Pipes-Munjal relationship, which is basically the Greenshield relationship, but the density ratio is raised to the power of n, where $n > 1$ and an integer. I allowed the user to also choose the value of n, further making what the code produces customizable.

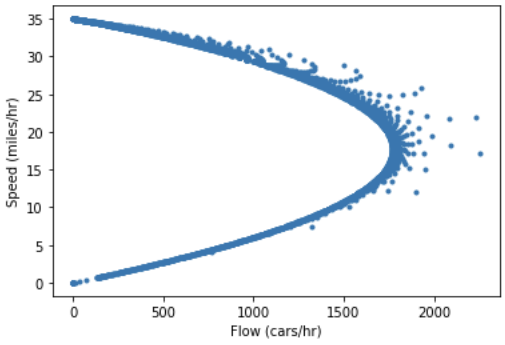
I also added charts to the code that plotted speed, density, and flow against each other. I did this by creating lists of speed and density as I was updating the density array. Knowing that flow $q = \rho v$, I could easily create a list for flow. Plotting these against each other showed the difference between the speed-density relationships. For example, the linear relationship of the Greenshield relationship and the exponential relationship of the Underwood relationship were evident. These plots could prove very useful in comparing the accuracy of different speed-density relationships, as seeing how many of the points on the graph fall onto the line of best fit at a specific speed limit/length of road being observed/time of traffic being observed could allow the user to select the best speed-density model for specific situations.

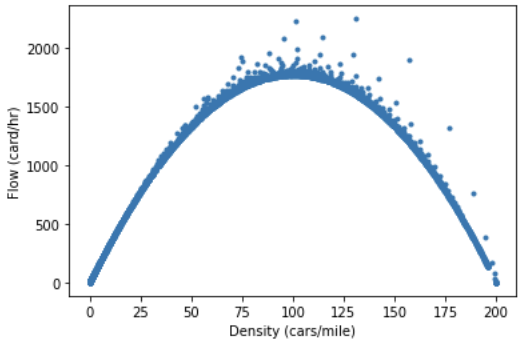
For example, here is the Greenshield relationship at 35 mph, 5 miles of road, and 5 minutes of a green light compared to the Underwood relationship:

```
Enter speed limit (mph):  35
Enter length of road we are observing (miles):  5
Enter duration that stoplight is green (minutes):  5
Enter speed-density relationship (greenshield, underwood, pipes-munjal):  greenshield
```

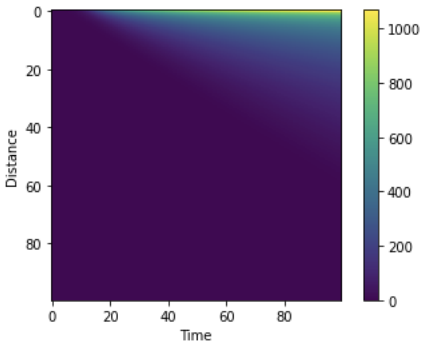


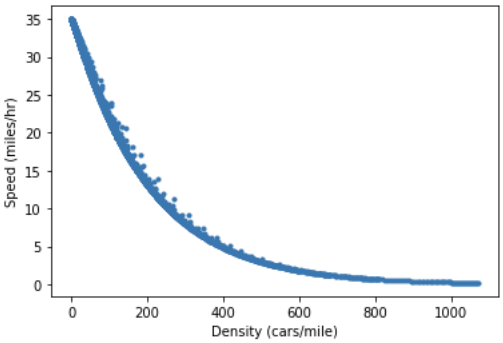


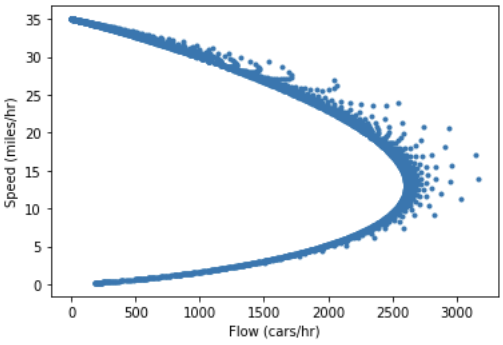


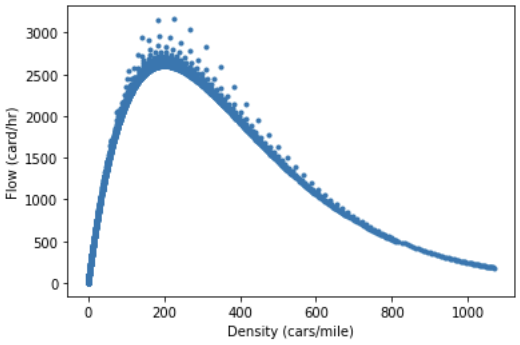


```
Enter speed limit (mph):  35
Enter length of road we are observing (miles):  5
Enter duration that stoplight is green (minutes):  5
Enter speed-density relationship (greenshield, underwood, pipes-munjal):  underwood
```



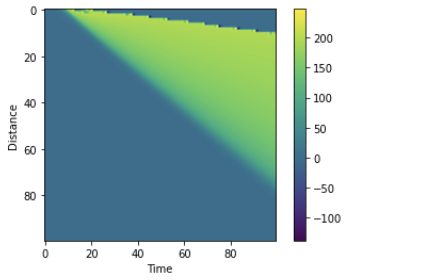


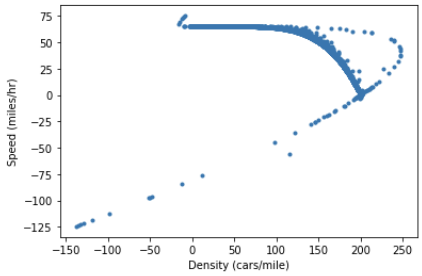


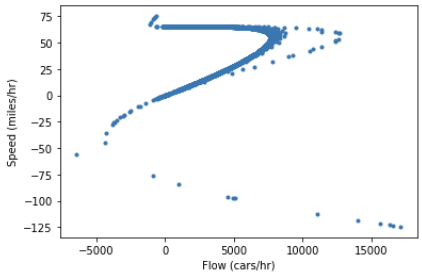


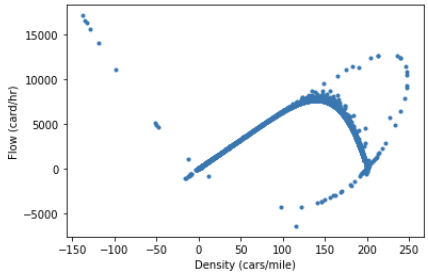
Another example of the plots created by the code: Pipes-Munjal:

```
Enter speed limit (mph):  65
Enter length of road we are observing (miles):  4
Enter duration that stoplight is green (minutes):  3
Enter speed-density relationship (greenshield, underwood, pipes-munjal):  pipes-munjal
Enter value of n for pipes-munjal model:  5
```









3 Final Comments

This might be the proudest I've ever been of a project because it actually works! I definitely did not realize how complicated traffic engineering was, as when I

first started the project, I thought I'd be able to compare efficiencies of different styles of intersections, but after seeing how complicated just this was, I cannot even imagine how complex created a model of a roundabout would be. This is by far the most difficult assignment I've completed at my time at Davidson, but I definitely think I have a much better understanding of computational and numerical methods after this, especially in terms of applying them to real world situations. I also have an unnecessarily extensive knowledge of traffic speed-density models now.

I used a macroscopic traffic model, which produces a very effective visual, but I do think it would be interesting to compare how a microscopic traffic flow model can present different information.

This project combined a lot of what I've learned in my differential equations class with computational methods, and despite this code being incredibly aggravating and frustrating, I could actually see myself maybe going into Urban Planning or Transportation Systems Engineering in grad school, because applying the math and computational methods I've learned to create something useful was very satisfying. I bet there would be some very fun and complicated equations when it came to modelling much more complex traffic.

4 References

Modelling Traffic Flow: Solving and Interpreting Differential Equations
Mathematical Modeling By Differential Equations
Modeling and Analysis of Traffic Flows using Four Dimensional Non-linear Dynamical System of Ordinary Differential Equations
Numerical Simulations of Traffic Flow Models
Macroscopic Traffic Flow Model
Modelling Traffic Flow with Partial Differential Equations
A Two-Dimensional Multi-Class Traffic Flow Model
Traffic Flow Simulation in Python
Notes on Traffic Flow
Flow, Speed, and Density
Adaptive numerical simulation of traffic flow density
A Numerical Study of Multi-class Traffic Flow Models
Numerical simulation of macroscopic traffic equations
Microscopic Traffic Flow Modeling
Microscopic Traffic Flow Model
An overview of microscopic and macroscopic traffic models
Traffic stream models
Speed-Density Relationship: from Deterministic to Stochastic