Traffic flow in a single and multiple lane highway with overtaking

Nikita Suprun

December 17, 2023

1 Simulation

1.1 Problem description

This project is about simulating traffic of a highway with overtakes and lane changes where no accidents on the road occur. We will consider one, two and three lane highways and study the flow rate (from definition: sum of car velocities per road length) with the help of a fundamental diagram (a plot of flow rate vs car density) for the different numbers of lanes and different maximum speeds.

Furthermore, we will consider and compare the results for lane change models used in two different regions:

- EU model: Left lane (from driver's perspective) is the overtaking lane, i.e. the cars only overtake on the left, and change to the right most available lane after the overtake.
- North American model: cars can overtake both on the left and on the right and do not necessarily change lane after the overtake.

Traffic of a highway is a complex system where each car driver has its own individual behaviour and is constantly interacting with the other drivers. There are three common methods of modelling traffic flow:

- Microscopic modelling: each individual agent (car driver) has its own behaviour, normally defined by the position and speed of the vehicles around it. Two dynamic processes have to be considered in this type of modelling: car-following and lane changing.
- Macroscopic modelling: all agents have the same behaviour. In this type of modelling, agents' behaviour is governed by averaging observables such as car density and, thus, the velocity is normally a function of car density: $v(\rho)$. This type of modelling is based suited for studying intersections.
- Maesoscopic modelling: this type of modelling is a mixture of microscopic and macroscopic modelling. Agents' behaviour is normally governed by the average car density in a specified region around the agent.

This simulation will focus on microscopic modelling, as we deem it to be the most suitable for studying the fundamental diagrams with lane changing and overtaking.

1.2 Model

1.2.1 Highway model

We start by determining how to model the highway. There are two approaches to consider: a discrete spatial model and a continuous spacial model.

In a discrete spacial model, each car would occupy a cell, a position in an array of size MxN representing the highway, where M is the number of lanes and N is the length of the highway. The speed, v and v_{safe} (the maximum speed that a car can go without a risk of crashing), also gain a discrete value.

The benefit of this model is ease of implementation and computational efficiency. We can for instance, precompute and store the possible values of v_{safe}. This approach also simplifies computing distances between cars and implementing lane change logic. However, the imposed restrictions on values of speed v and v_{safe} will affect cars' behaviour in traffic jams. The cars will namely start breaking earlier as the next cell might be too close to the car in front. This leads to loss of crucial information and fundamental diagram being dependent on which discretisation is chosen.

To avoid the consequences of discretisation, we will choose a suitable size for the array such that the discretisation effects become negligible.

1.2.2 Boundary conditions

Now we need to decide between modelling the highway as a closed loop or a slice which the cars enter at a certain rate. The slice model introduces additional complexity to the simulation, due to an additional parameter of rate of cars entering the slice.

In this simulation, we will use periodic boundary conditions. This means, the road is modelled as a closed loop where the cars are going around.

1.2.3 Initial conditions

The initial conditions we use in the simulation are that cars are randomly positioned across the highway and start as stationary. We will ensure the steady state of the simulation before recording the traffic flow for the fundamental diagram.

1.2.4 Modelling car behaviour

We introduce the following rules that will govern the simulation:

- Speed limit for each individual car: $v_i(t) \leq v_{max}(i)$, where $v_{max}(i)$ is the maximum allowed speed for car i.
- Limit acceleration by some value $a_{max} > 0$: $v_i(t+\Delta t) \leq v_i(t) + a_{max}\Delta t$.
- Introduce safe speed: $v_{safe} = d(i,i+1) - 1$, where $d(i,i+1)$ is the distance between car i and the car in front. This is done to avoid collisions between the cars.
- Introduce a probability, p_s, a car will decelerate in a given tick. This can be translated into drivers not being able to keep constant speed in real life. This will also make the simulation less deterministic!
- Lane change rules:
  - NA model: A lane change will occur to the random close by free lane, if the car cannot accelerate or keep its maximum speed. Free lane implies the following condition: $d(x_i, x_{i-1}) > v_{max}(i-1)$, i.e. the safe distance to the car behind is fulfilled.
  - EU model: A lane change will occur to the left lane, given it is free, if the car cannot accelerate or keep its maximum speed. The lane change to the right most available lane will occur, if the car can keep accelerating or going at its maximum speed.

  See the Appendix section with code listings and the full logic attached for more details!

We take the following numerical values of the parameters that are kept constant throughout one simulation:

- $a_{max} = 1$ $[cell/tick^2]$.
- $\Delta t = 1$ $[tick]$. The model is iterative, and each iteration is a new tick.
- $p_s = 0.2$. We assume there is a 20% change a car's speed will decrease its speed within one minute. We choose a small value to only have a slight disturbance in the simulation.
- L $[cells]$ will be chosen for each simulation separately. For each case we will choose L large enough such that the fundamental diagram becomes independent of highway length.
- T $[ticks]$ - the simulation run time will be chosen for each simulation separately, such that the traffic flow has reached steady-state.

The simulation can be formulated by the following algorithm:

1. Update the lanes for all cars simultaneously according to the logic described above, based the current state of the other cars.

2. Update the speed of each car simultaneously:

   (a) Rule 1: If $v < v_{max}$, increase v by 1 cell/tick.

   (b) Rule 2: If distance, $v_i \leq d(i, i+1)$, then reduce v_i to $v_i = d(i, i+1) - 1$ cells/tick.

   (c) Rule 3: Reduce v_i by 1 cell/tick, with probability $p_s = 0.2$, given $v_i > 0$.

3. Update the positions of the cars: $x_i(t+1) = x_i(t) + v_i(t)$.

Hence, in our simulation we assume that the lane change occurs in one time step. The implementation of the simulation is attached as code listings to the Appendix section.

2 Problem

Using the model described above, we will investigate the traffic flow with the help of fundamental diagram when varying the number of lanes and lane change model, as well as maximum speed, v_{max}, of the cars and consider different distributions of it. We will attempt to answer the following questions:

- How does the traffic flow depend on the number of lanes, when v_{max} is the same for all cars? Is the dependency different when v_{max} is stochastic?

- What quantitative differences in traffic behaviour can be observed between NA and EU lane change traffic models?

- How many lanes and what lane change model should one use to increase the traffic flow. Does the answer depend on the expected car density on the road?

3 Results and analysis

3.1 Case 1: Same v_{max} for all cars on the highway

In this section we will consider the following values of v_{max}: $v_{max} = \{2, 5, 10\}$ $[cells/tick]$.

To ensure the statistical accuracy of the results, we determine a suitable N (number of runs) for each simulation. As we observe in Figure 1, in the particular case of 3 lane highway with 100 cells and density of 0.3 with NA lane change model, the standard error decreases fast initially and we get standard error below 0.01 after 3 runs. We also observe that the standard error is similar for all values of v_{max}. All future figures are produced by running the simulation 5 number as it gives a good balance between the simulation runtime and results in the standard error of about 0.005.

In Figure 2, we can observe a plot of flow rate against time and see that the settling time, i.e. the time to reach steady state, is below 500 ticks, when considering density of 0.3 and 100 cells as road length. We will proceed by running the simulation for 500 ticks and take the average flow rate of the last 100 ticks.

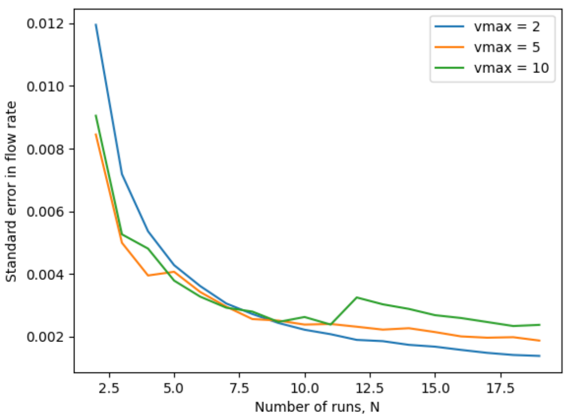

Figure 1: Standard error in flow rate against number of runs N, recorded with T=500 ticks, cars density = 0.3 cars/road length, road length of 100 cells, 3 lanes and NA lane change model.

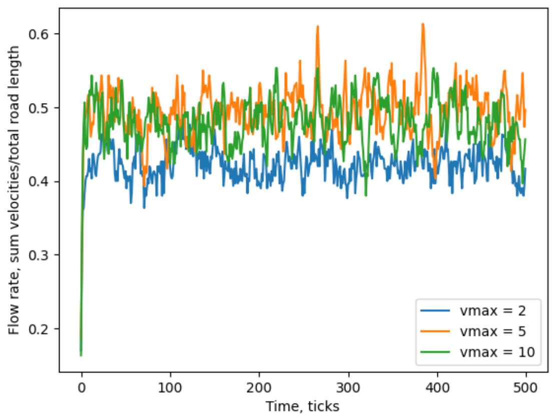

Figure 2: Flow rate against time, recorded with T=500 ticks, cars density = 0.3 cars/road length, road length of 100 cells, 3 lanes and NA lane change model.

Lastly, we ensure the chosen length of 100 cells is suitable for plotting the fundamental diagram and the effects of discretisation can be neglected. In Figure 3, we observe an example of how the fundamental diagram looks like for different road lengths. As we see, for lengths larger than 100 cells, the fundamental diagram becomes convergent to that of the continuous highway model. We proceed with road length = 100 cells!

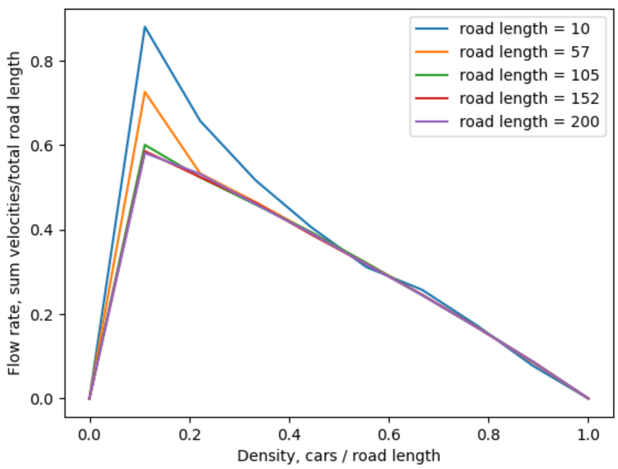

Figure 3: Fundamental diagram with v_{max} = 10 cells/tick, 3 lanes and NA lane change model.

We will now study how the traffic flow is affected when considering different maximum speeds, number of lanes and lane change models. In Figure 4 we see the fundamental diagram with v_{max} = 10 cells/tick and Figure 5, with v_{max} = 2 cells/tick for different number of lanes and lane change model. We can draw some conclusions from the diagrams:

1. The fundamental diagram is independent of the number of lanes and lane change model considered when all cars have the same v_{max}. This has been confirmed by running the simulation for other values of v_{max}.
2. The peak flow rate is higher for higher values of v_{max}, for instance about 0.7 cars/road length for v_{max} = 10 cells/tick vs about 0.5 cars/road length for v_{max} = 2 cell/tick. Additionally, the peak occurs at lower density for higher v_{max} values.

Independence on number of lanes is somewhat expected, as all the cars having the same v_{max} means there is no bottle neck from slower cars that would cause the faster cars to slow down. Furthermore, when there are more lanes available, the cars try to distribute themselves in a way that maximizes the flow rate. As there is a probability of 20% a car will decelerate after one tick, the slightly higher flow rate for 2 lane EU lane change model in Figure 4 (OBS: the flow rate is the lowest for 1 lane highway) and 3 lane EU lane change model in Figure 5 can be explained by the fact that car i can avoid the need to break if the car i+1 (the car in front) breaks, if there is a lane available they can change to and keep their speed/accelerate.

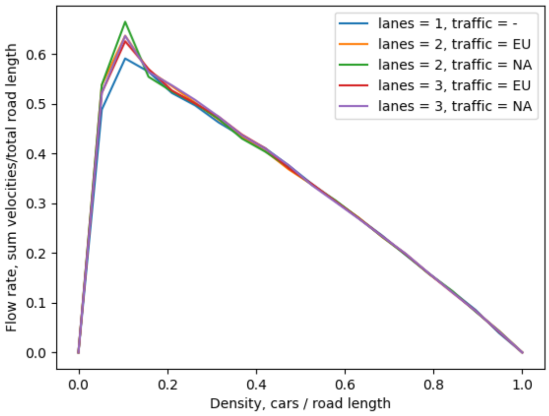

Figure 4: Fundamental diagram with v_{max} = 10 cells/tick, for 1, 2 and 3 lanes with different lane change models.

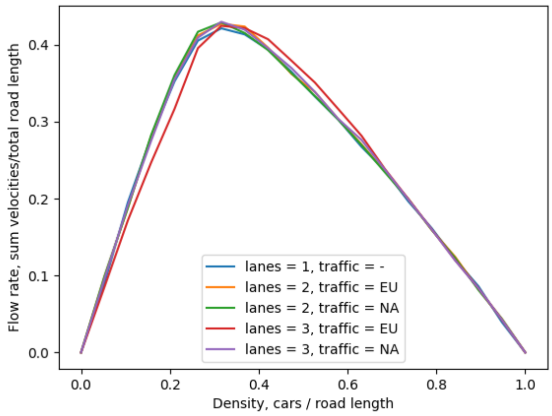

Figure 5: Fundamental diagram with v_{max} = 2 cells/tick, for 1, 2 and 3 lanes with different lane change models.

Another interesting property one can study, is the number of cars passing through each cell in the highway. Figure 6 shows the heat map for 3 lane NA lane change model, while Figure 7 shown the heat map for 3 lane EU lane change model. Both heat maps were produced with v_{max} = 5 cells/tick and density = 0.1 cars / cell. We make the following observations:

1. The load (cars passing through each cell) appears to be somewhat evenly distributed in the NA lane change model.
2. The cars tend to stick to the right most lane (lane 0 in Figure 7) in the EU lane change model, which increases the load on the right and the middle lane, however keeps the left most lane (lane 2) free.

This is expected behaviour! We can also draw conclusions that lane changes in NA lane change model are more stochastic, as the cars might change both to the right and left lane to overtake, rather than only to the left lane as in the EU model.

As we increase the density, the number of cars using the middle and the left lane increases in the EU lane change model, while the load remains somewhat uniform in the NA lane change model.

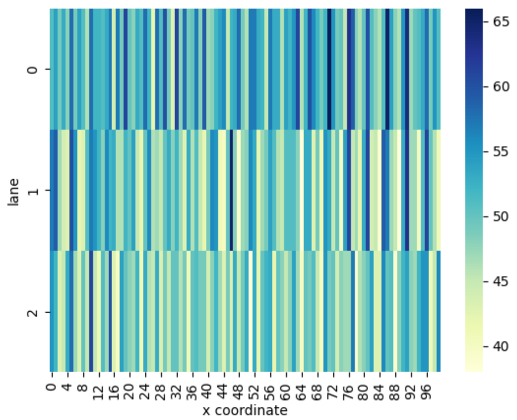

Figure 6: Heat map showing how many cars pass through each cell, 3 lane highway NA lane change model.

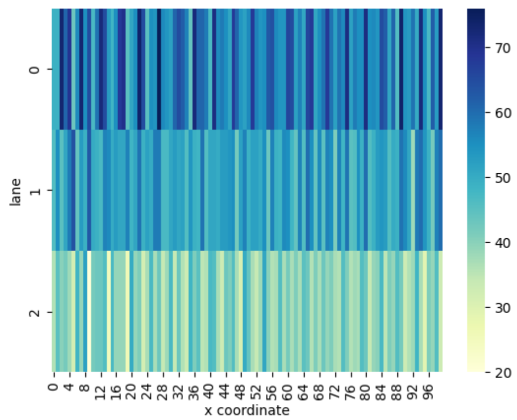

Figure 7: Heat map showing how many cars pass through each cell, 3 lane highway EU lane change model.

3.2 Case 2: Stochastic v_{max} for each car on the highway

Now, we will consider two different distributions for v_{max}, in particular:

- Normal distribution with different σ (standard deviation) and μ (mean) parameters.
- Uniform distribution with different boundary values: [a, b] - a lower boundary, b upper boundary.

In Figure 8, we show the standard error dependency on number of runs, N, for different stochastic models of v_{max} for a 3 lane highway. As we see, the standard error is about the same for the different lane change and v_{max} distribution models and the dependency on N is of the same nature as in Case 1. We further observe that the standard error is larger than in Case 1, which is expected as the traffic is now more stochastic, and reaches the value of about 0.002 across all models at N = 20. We increase N to 10, as larger increments result in run time that is too long and proceed with the study!

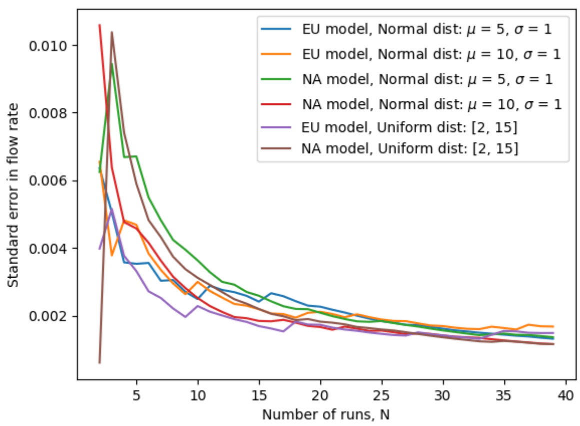

Figure 8: Standard error in flow rate against number of runs N, recorded with T=500 ticks, cars density = 0.3 cars/road length, road length of 100 cells, 3 lanes.

In Figure 9 we show an example of the fundamental diagram for v_{max} drawn from normal distribution and from uniform distribution in Figure 10. We observe a number of differences from Case 1, where v_{max} was the same for all cars on the highway:

1. For certain densities, the range of about 0.1-0.4 cars/cell, the fundamental diagram is no longer independent of the number of lanes and the lane change model considered. This is reasonable, consider an

extreme case of 1 lane highway with one car that has v_{max} = 2 cells/tick and the rest of the cars having v_{max} = 10 cells/tick: all the cars will pile up behind the slow car and the flow rate will be low. This situation would have been avoided if an additional lane was added!

When the density is high enough, there is not enough free space for the cars to reach their v_{max}, which becomes the limiting factor in flow rate. Hence, the threshold density is about 0.4 (in cases shown in Figure 9 and Figure 10, generally this value can be different) and the fundamental diagram starts to become independent of the number of lanes and lane change model.

2. When v_{max} is normally distributed (most realistic case in highway traffic), there is a slight difference in flow rate between the different highways. In our case (μ = 10, σ = 4), 68.2% of cars had speed in the range of [6, 14]. This resulted in higher flow rate in 3 lane and 2 lane highway with EU lane change model. The 2 and 3 lane highway with NA lane change model performed worse, which could be explained by the fact that lane changes are more stochastic in NA lane change model and EU lane change model is more algorithmic and hence, optimal.

3. When v_{max} is uniformly distributed with a broad range, the number of lanes has a larger effect on flow rate than in other v_{max} distribution models. As we see, when there is a larger spread of v_{max}, the performance of 1 lane highway decreases significantly (See the flow rate of 1 lane highway compared to 3 lane highways in Figure 9 vs Figure 10). Here, the 3 lane highway with EU lane change model showed the best result again!

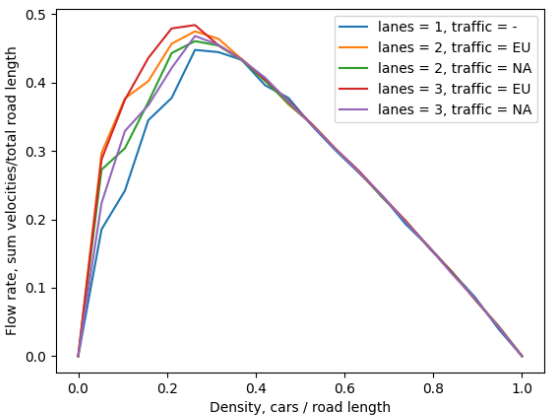

Figure 9: Fundamental diagram with v_{max} drawn from normal distribution, μ = 10, σ = 4.

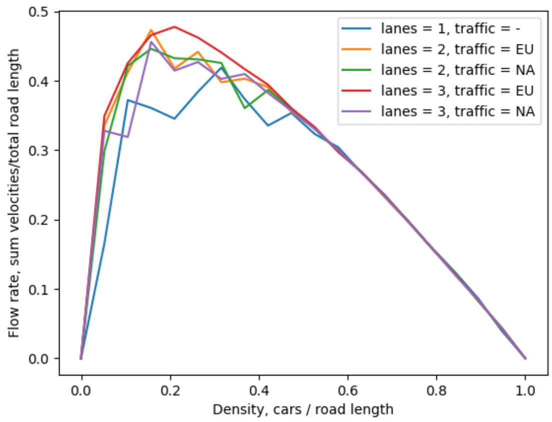

Figure 10: Fundamental diagram with v_{max} drawn from uniform distribution, [2, 15].

We attempted to adjust the value of σ and μ in the normal distribution and came to the conclusion that increasing standard deviation σ and/or lowering μ causes a larger spread of flow rate between the different number of lanes. Setting σ to 0 results in the same fundamental diagrams as those seen in Case 1, as expected!

We also attempted to adjust the boundaries in the uniform distribution and observed that smaller range results in smaller spread of flow rate across the different number of lanes in the highway.

4 Conclusion

All in all, we studied the traffic flow for 1, 2 and 3 lane highways with 2 different lane changing models and different distribution of v_{max} for the cars.

We arrive at the conclusion that the fundamental diagram and hence, the traffic flow, is independent of the number of lanes or lane change model when v_{max} is the same for all cars. When v_{max} is stochastic, this changes: we observed that the number of lanes and lane change model has an effect on the flow rate. 3 lane highway with EU lane change model had the highest flow rate for both normal and uniform distribution of v_{max}, while 1 lane highway had the lowest flow rate in both cases.

The larger the spread of v_{max}, the higher the different in flow rate between the different number of lanes. The effect of additional lanes is more visible with uniform distribution of v_{max} with a broad range of boundaries.

We also observed quantitative differences in traffic behaviour between EU and NA lane change model: EU lane change model has shown to be more effective in utilizing the additional lanes and leading to a higher flow rate when v_{max} is stochastic. The different lane change model also affects how cars spread out across different lanes. The distribution was somewhat uniform for NA lane change model and the right most lane was the most busy in EU lane change model.

Finally, we observed that at some car density, the effect of additional lanes was diminished and the fundamental diagram became independent of number of lanes or lane change model. For the parameters we studied, this density was about 0.4 cars/cell.

5 Appendix: Python code for the simulation

```python
class Road:
    """ Class for the state of a number of cars """

    def __init__(self, density, road_length, mu, sigma=1, num_lanes=1):
        self.road_length = road_length
        self.num_lanes = num_lanes
        self.num_cars = np.round(density * road_length * num_lanes)
        self.t = 0
        self.road = [[[-1]*2] * road_length for _ in range(num_lanes)]
        counter = 0
        while counter < self.num_cars:
            lane = random.randrange(0, num_lanes)
            x = random.randrange(0, road_length)
            if self.road[lane][x][0] == -1:
                v_max = np.random.randint(2, 15) if mu == -1 else np.random.normal(mu, sigma)
                v_max = int(np.round(v_max))
                self.road[lane][x] = [0, v_max if v_max > 0 else 1]
                counter += 1

    def distance(self, direction, car_x, lane_to_check, limit):
        offset = 0
        while offset < limit and self.road[lane_to_check][(car_x + direction * offset) % self.road_length][0] == -1:
            offset += 1
        return offset


class Propagator:
    def __init__(self, lane_change_model):
        self.lane_change_model = lane_change_model
        self.p_s = constants.p_s

    def is_lane_available(self, cars, lane, x, v, v_max):
        d_ahead = cars.distance(1, x, lane, v_max + 2)
        d_behind = cars.distance(-1, x, lane, v_max + 2)
        return (d_ahead >= v + 2) and (d_behind >= v_max + 2)

    def update_lanes(self, cars):
        if cars.num_lanes == 1:
            return

        road_updated = [[[-1]*2] * cars.road_length for _ in range(cars.num_lanes)]
        for lane in range(cars.num_lanes):
            for x in range(cars.road_length):
                if cars.road[lane][x][0] != -1:
                    d = cars.distance(1, x + 1, lane, cars.road[lane][x][1] + 2) + 1
                    left_lane = lane + 1
                    right_lane = lane - 1

                    if self.lane_change_model == "EU":
                        if right_lane >= 0 and self.is_lane_available(cars, right_lane, x, cars.road[lane][x][0], cars.road[lane][x][1]):
                            road_updated[right_lane][x] = cars.road[lane][x]
                        elif d < cars.road[lane][x][0] + 2 and left_lane < cars.num_lanes \
                                and self.is_lane_available(cars, left_lane, x, cars.road[lane][x][0], cars.road[lane][x][1]):
                            road_updated[left_lane][x] = cars.road[lane][x]
                        else:
                            road_updated[lane][x] = cars.road[lane][x]

                    elif self.lane_change_model == "NA":
                        if d < cars.road[lane][x][0] + 2:
```

```
                            random_lanes = [right_lane, left_lane]
                            random.shuffle(random_lanes)
                            if 0 <= random_lanes[0] < cars.num_lanes \
                                    and self.is_lane_available(cars, random_lanes[0], x, cars.road[lane][x][0], cars.road[lane][x][1]):
                                road_updated[random_lanes[0]][x] = cars.road[lane][x]
                            elif 0 <= random_lanes[1] < cars.num_lanes \
                                    and self.is_lane_available(cars, random_lanes[1], x, cars.road[lane][x][0], cars.road[lane][x][1]):
                                road_updated[random_lanes[1]][x] = cars.road[lane][x]
                            else:
                                road_updated[lane][x] = cars.road[lane][x]
                        else:
                            road_updated[lane][x] = cars.road[lane][x]
                    else:
                        raise "The lane change model chosen is incorrect. Should be one of: NA, EU."

        cars.road = road_updated

    def update_speed(self, cars):
        for lane in range(cars.num_lanes):
            for x in range(cars.road_length):
                v_max = cars.road[lane][x][1]
                if cars.road[lane][x][0] != -1:
                    d = cars.distance(1, x + 1, lane, v_max + 2) + 1

                    # Rule 1: is v < v_max, increase v by 1 unit
                    if cars.road[lane][x][0] < v_max:
                        cars.road[lane][x][0] += 1

                    # Rule 2: If dist, d, to next car is <= v_i, reduce v_i to d - 1
                    if d <= cars.road[lane][x][0]:
                        cars.road[lane][x][0] = d - 1

                    # Rule 3: Reduce v_i by 1 unit, with probability p_s, given v_i>0
                    if cars.road[lane][x][0] > 0 and np.random.rand() < self.p_s:
                        cars.road[lane][x][0] -= 1

    def update_position(self, cars):
        sum_velocities = 0
        road_updated = [[[-1]*2] * cars.road_length for _ in range(cars.num_lanes)]
        for lane in range(cars.num_lanes):
            for x in range(cars.road_length):
                if cars.road[lane][x][0] != -1:
                    x_next = (x + cars.road[lane][x][0]) % cars.road_length
                    road_updated[lane][x_next] = cars.road[lane][x]
                    sum_velocities += cars.road[lane][x][0]
        cars.road = road_updated
        return sum_velocities

    def timestep(self, cars):
        self.update_lanes(cars)
        self.update_speed(cars)
        sum_velocities = self.update_position(cars)

        cars.t += 1

        return sum_velocities / (cars.road_length * cars.num_lanes)  # return the flow for the entire highway

    # Plotting observables and analysis
```