

MA5710 - Assignment 4

Abhinav T K, MA23M002

November 5, 2023

1. Implement Gudanov Method to numerically solve the first order hyperbolic partial differential equation, that is the scalar conservation law. Assume different speed density relations and initial values as mentioned in the slide provided, on case by case basis(when the signal is red, then red to green and then red again, with and without speed breaker before and after the signal). Generate the tables provided in the slide for each case and subcases.

Code: **traffic_flow_model.m**

```
1 % Main file
2 clear;
3 clc;
4
5 rhoMax = 1;      % Maximum rho
6 rhoMin = 0;      % Minimum rho
7 h = 0.1;        % space step size
8 k = 0.1;        % time step size
9 x = -1:h:1;      % Road - (-1 to 1)
10 x_divs = length(x); % No. of space steps
11
12 total_time= 3;    % Total time under consideration
13 t_steps = total_time / k; % No. of time steps
14
15 % Initial conditions
16 rho0 = 0.65 * ones(1, x_divs);
17
18 u = zeros(t_steps, x_divs); % Array to store u
19
20 % Initial and boundary conditions for u
21 u(1,:) = 1 - 2 .* rho0/rhoMax ; % Burger's equation
22 u(:,1) = u(1,1)*ones(t_steps,1);
23
24 % Signal
25 x_signal = 0.2; % Position of signal
26 x_signal_idx = floor((x_signal - x(1))/h); % Index of signal position in x array
27
28 signal_times = [0.5, 1.5, 2.5]; % Signal timings (in min.)
29 signal_time_idx = signal_times/k; % Index of signal time in time array
30
31
32 f = @(u) (u.^2)/2; % Flux function
33 df = @(u) u; % Derivative of Flux
34
35 % Apply boundary condition (1)
36 for i = 1:2:length(signal_time_idx)-1
37     u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx) = -1;
38     u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx + 1) = 1;
39 end
40
```

```

41 % Numerical calculation using Godunov method
42 for t = 1:t_steps-1
43     % Apply boundary condition (2)
44     for i = 1:2:length(signal_time_idx)-1
45         u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx) = -1;
46         u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx + 1) = 1;
47     end
48
49     for i = 2:x_divs - 1
50         u(t+1,i) = u(t,i) - (k/h)*(f(u_star(t,i, u,f, df))-f(u_star(t,i-1, u,f, df))
51             );
52     end
53 u(:,1) = u(:,2);
54 u(:,end) = u(:,end-1);
55
56 % Get rho value from u
57 rho_tab = (rhoMax / 2).*(1 - u);
58
59
60 % Plotting at different cases
61 figure;
62 % First subplot - when green light (initial cond.)
63 subplot(2, 2, 1);
64 plot(x, rho_tab(1,:), 'LineWidth', 1.5);
65 % Add labels and title
66 xlabel('Road');
67 ylabel('Traffic density');
68 title('t = 0, Uniform traffic flow (Green signal)');
69 % Display grid
70 grid on;
71 hold on;
72 % Add a vertical line at signal position
73 xline(x_signal, 'g--', 'LineWidth', 2); % Signal position
74 hold off;
75
76 % Second subplot - when red light
77 subplot(2, 2, 2);
78 plot(x, rho_tab(10,:), 'LineWidth', 1.5);
79 % Add labels and title
80 xlabel('Road');
81 ylabel('Traffic density');
82 title('During Red signal');
83 grid on;
84 hold on;
85 % Add a vertical line at signal position
86 xline(x_signal, 'r--', 'LineWidth', 2);
87 hold off;
88
89 % Third subplot - when red light (just before green light)
90 subplot(2, 2, 3);
91 plot(x, rho_tab(15,:), 'LineWidth', 1.5);
92 % Add labels and title
93 xlabel('Road');
94 ylabel('Traffic density');
95 title('Just before Green signal');
96 grid on;
97 hold on;
98 % Add a vertical line at signal position

```

```

99 xline(x_signal, 'r--', 'LineWidth', 2);
100 hold off;
101
102 % forth subplot
103 subplot(2, 2, 4);
104 plot(x, rho_tab(25,:), 'LineWidth', 1.5);
105 % Add labels and title
106 xlabel('Road');
107 ylabel('Traffic density');
108 title('Traffic flow 1 min after signal turned green');
109 grid on;
110 hold on;
111 % Add a vertical line at signal position
112 xline(x_signal, 'g--', 'LineWidth', 2);
113 hold off;
114
115
116
117 % Function for u_star calculation used in Godunov scheme
118 function u_star_out = u_star(t, i, u, f, df)
119     gradF = df(u(t, i)); % Gradient of flux for u(t,i)
120     gradF1 = df(u(t, i + 1)); % Gradient of flux for u(t,i+1)
121     % Apply different conditions...
122     if gradF >= 0 && gradF1 >= 0
123         u_star_out = u(t, i);
124     elseif gradF < 0 && gradF1 < 0
125         u_star_out = u(t, i + 1);
126     elseif gradF >= 0 && gradF1 < 0
127         s = (f(u(t, i + 1)) - f(u(t, i))) / (u(t, i + 1) - u(t, i));
128         if s >= 0
129             u_star_out = u(t, i);
130         else
131             u_star_out = u(t, i + 1);
132         end
133     else
134         u_star_out = 0;
135     end
136 end

```

Output:

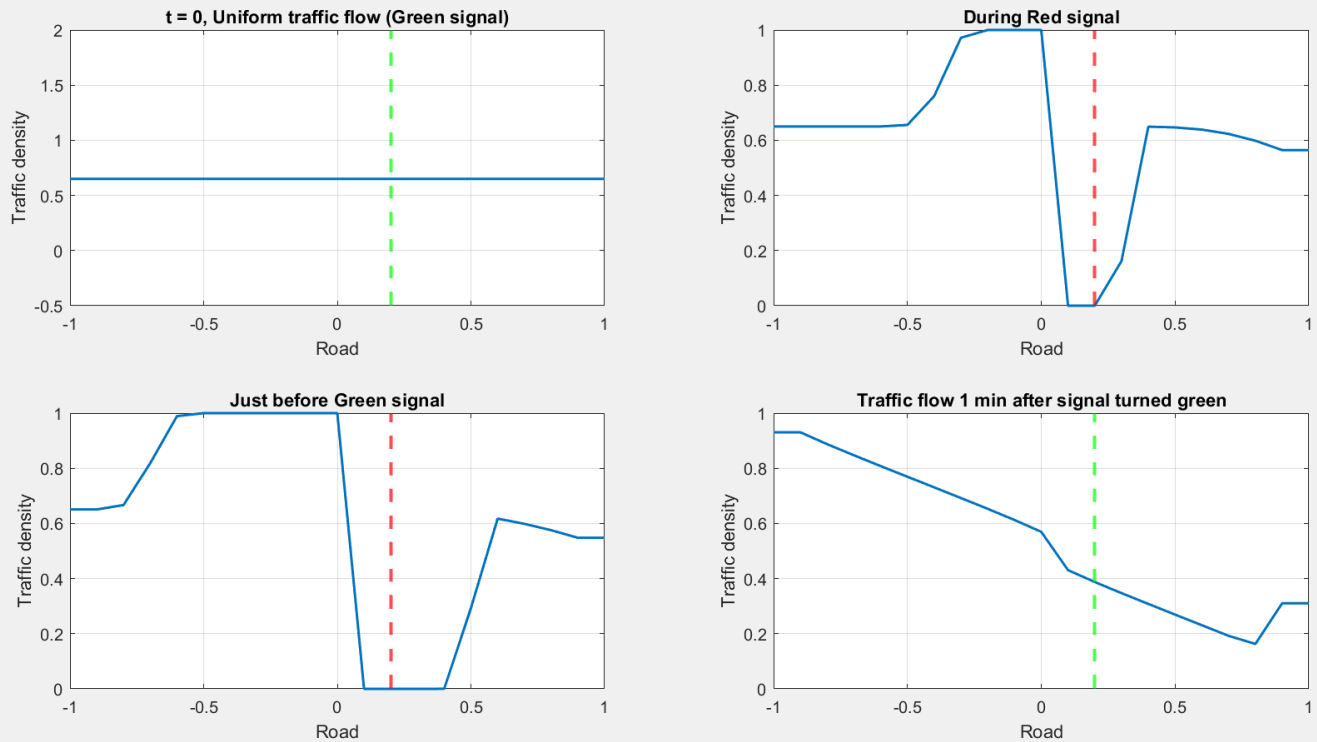


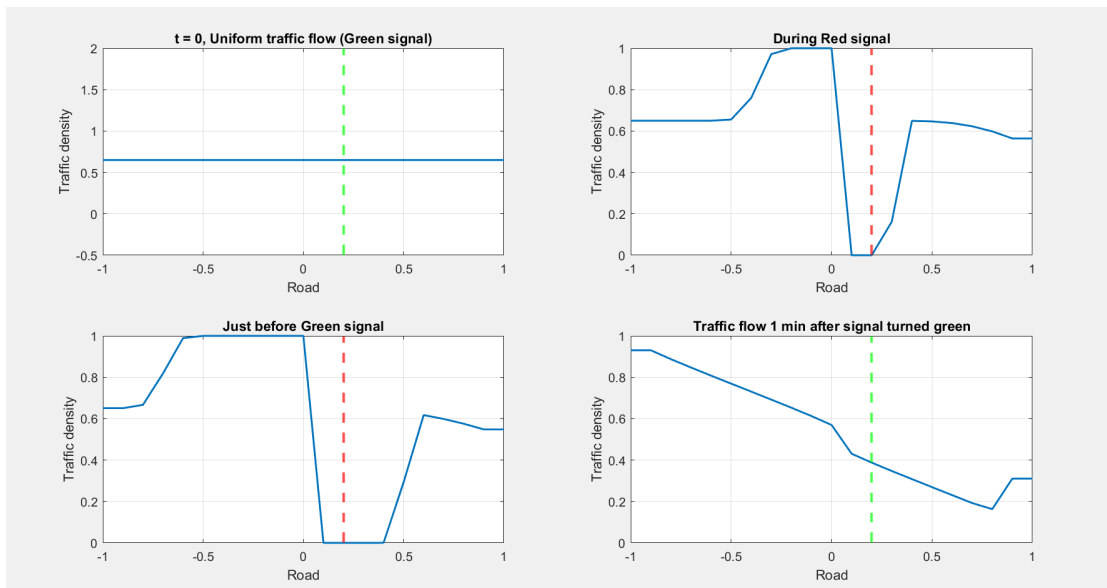
Figure 1: Density vs Road length for different traffic signal conditions

t/x	Traffic signal position																					
	-1	-0.9	-0.8	-0.7	-0.6	-0.5	-0.4	-0.3	-0.2	-0.1	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1	
0	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
0.1	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
0.2	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
0.3	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
0.4	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
0.5	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.8775	1	0	0.4225	0.65	0.65	0.65	0.649453	0.644564	0.62689	0.589094	
0.6	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.770006	0.984994	1	0	0.195	0.65	0.65	0.65	0.649836	0.639766	0.618726	0.581156	
0.7	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.700403	0.932322	0.999775	1	0	0.038025	0.579475	0.644951	0.649294	0.645641	0.634327	0.611217	0.57457	
0.8	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.667662	0.847144	0.95195	1	0	0.001446	0.388539	0.649754	0.648217	0.642474	0.628653	0.604408	0.569009	
0.9	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
1	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
1.1	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
1.2	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
1.3	0.650047	0.650047	0.652004	0.694298	0.902533	0.98618	1	1	1	1	1	1	0	0	0.000696	0.353315	0.641609	0.630304	0.612317	0.587754	0.556505	
1.4	0.650638	0.650638	0.666651	0.818579	0.98912	0.999998	1	1	1	1	1	1	0	0	0.4847	0.125527	0.637072	0.62594	0.607403	0.583246	0.553312	
1.5	0.655719	0.655719	0.740371	0.956325	0.9998	1	1	1	1	1	1	0.75	0.25	0	0	0.000248	0.293002	0.617392	0.598496	0.575439	0.547923	
1.6	0.689248	0.689248	0.890825	0.997972	1	1	1	1	1	1	0.8125	0.6875	0.3125	0.1875	0	0	0.617E-08	0.086098	0.584245	0.594486	0.572045	
1.7	0.806178	0.806178	0.986057	0.999996	1	1	1	1	1	0.847656	0.75	0.652344	0.347656	0.25	0	0.50E-05	0.187451	0.587265	0.56608	0.544618	0.541648	
1.8	0.948685	0.948685	0.998001	1	1	1	1	0.870865	0.789291	0.710709	0.629135	0.370865	0.289291	0.210709	0.129135	0.302E-09	0.035193	0.493945	0.56608	0.544618	0.541648	
1.9	0.997168	0.997168	1	1	1	1	1	0.887541	0.817014	0.75	0.682986	0.612459	0.387541	0.317014	0.25	0.182986	0.112459	0.001239	0.281925	0.561015	0.538321	
2	0.999992	0.999992	1	1	1	1	0.900188	0.837851	0.779016	0.720894	0.662149	0.599812	0.400188	0.337851	0.279016	0.220894	0.162149	0.099814	0.080719	0.514297	0.368582	
2.1	1	1	1	1	1	0.91015	0.854181	0.801557	0.75	0.698443	0.645819	0.58985	0.41015	0.354181	0.301557	0.25	0.198443	0.14582	0.096366	0.340489	0.353494	
2.2	1	1	1	1	1	0.918223	0.863731	0.819674	0.773121	0.726879	0.680326	0.632659	0.581777	0.418223	0.363731	0.319674	0.273121	0.226879	0.180326	0.133843	0.051005	
2.3	1	1	1	1	0.924911	0.878274	0.834601	0.792077	0.75	0.707923	0.665399	0.621726	0.575089	0.424911	0.378274	0.334601	0.292077	0.25	0.207923	0.165723	0.157143	
2.4	0.930549	0.930549	0.887453	0.84714	0.807952	0.769268	0.730732	0.692048	0.65286	0.612547	0.569451	0.40549	0.387453	0.34714	0.303955	0.269268	0.230732	0.192155	0.162953	0.140046	0.31046	
2.5	0.895296	0.895296	0.85784	0.821468	0.785623	0.75	0.714377	0.67852	0.64216	0.606474	0.564627	0.435373	0.395296	0.35784	0.321468	0.285623	0.25	0.214418	0.181785	0.232751	0.223751	
2.6	0.867086	0.867086	0.833132	0.799707	0.766542	0.733458	0.700293	0.666868	0.632914	0.599198	0.560451	0.439549	0.402082	0.367086	0.333132	0.299707	0.266542	0.233475	0.201489	0.202912	0.202912	
2.7	0.843311	0.843311	0.811979	0.780928	0.75	0.719072	0.688021	0.656889	0.624836	0.593918	0.556796	0.43204	0.408016	0.375164	0.343311	0.311979	0.280928	0.250008	0.219562	0.202064	0.202064	
2.8	0.82278	0.82278	0.793569	0.764507	0.735493	0.706431	0.67722	0.647721	0.617713	0.586749	0.553571	0.446429	0.413251	0.382827	0.352729	0.32278	0.293569	0.264511	0.235712	0.221518	0.221518	
2.9	0.804776	0.804776	0.77735	0.75	0.72625	0.695224	0.667635	0.639756	0.611382	0.582903	0.550701	0.449259	0.417907	0.388618	0.360244	0.332365	0.304776	0.277352	0.250105	0.225174	0.225174	

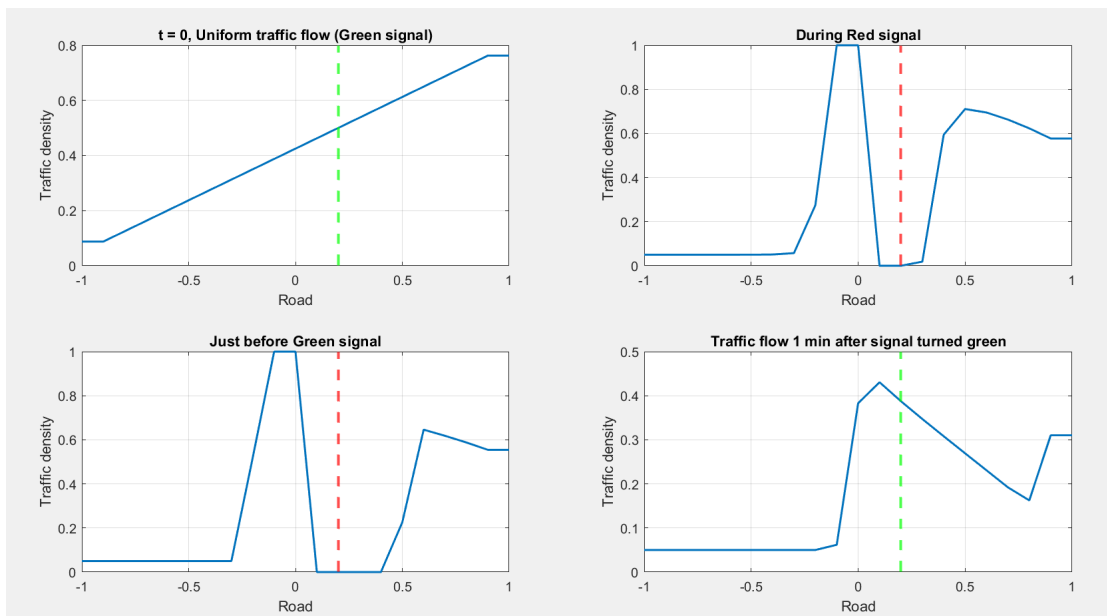
Figure 2: Output Density data

Different initial conditions (Comparisons):

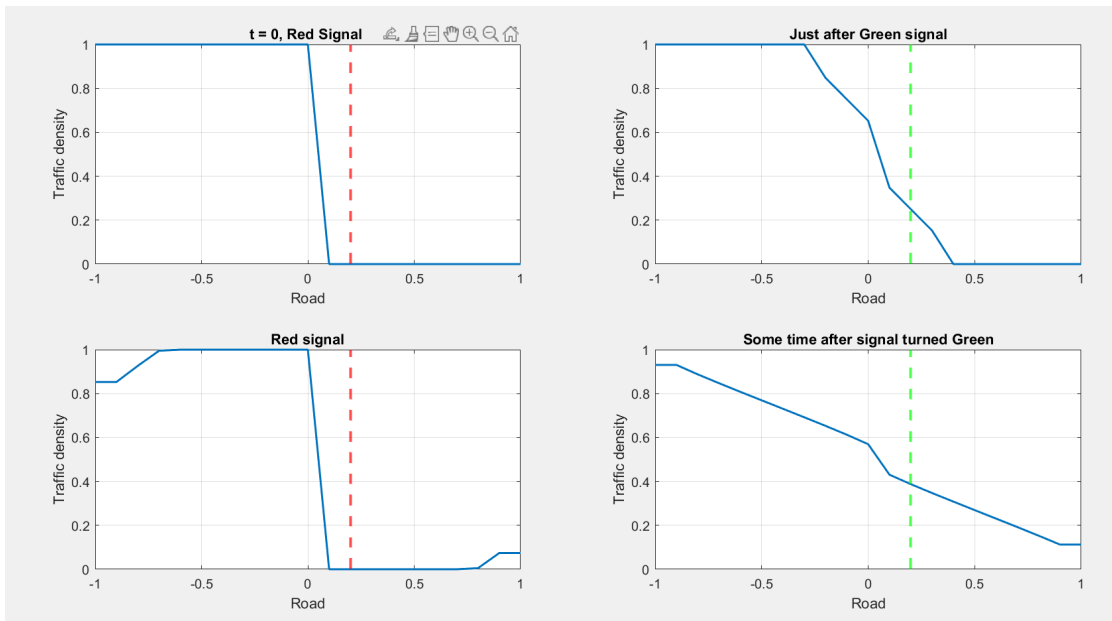
1. $\rho_0 = 0.65 * \text{ones}(1, x_divs)$



2. $\rho_0 = \text{linspace}(0.05, 0.8, x_divs)$



3. $\rho_0(x \leq 0) = 1$; $\rho_0(x > 0) = 0$;



Traffic flow with speed breaker before signal:

Code: traffic_flow_model_spdbrk_left.m

```
1 % Traffic flow model with speed breaker before signal
2 clear;
3 clc;
4
5 rhoMax = 1;      % Maximum rho
6 rhoMin = 0;      % Minimum rho
7 h = 0.1;         % space step size
8 k = 0.1;         % time step size
9 x = -1:h:1;      % Road - (-1 to 1)
10 x_divs = length(x); % No. of space steps
11
12 total_time= 3;    % Total time under consideration
13 t_steps = total_time / k; % No. of time steps
14
15 % Initial conditions
16 rho0 = 0.65 * ones(1, x_divs);
17
18 % rho0 = np.linspace(0.1, 0.5, num_divs)
19 % rho0 = np.zeros(num_divs)
20 % rho0[x <= 0] = 1
21 % rho0[x > 0] = 0
22
23 u = zeros(t_steps, x_divs); % Array to store u
24
25 % Initial and boundary conditions for u
26 u(1,:) = 1 - 2 .* rho0/rhoMax ;
27 u(:,1) = u(1,1)*ones(t_steps,1);
28
29 % Signal
30 x_signal = 0.2; % Position of signal
31 x_signal_idx = floor((x_signal - x(1))/h); % Index of signal position in x array
32
33 signal_times = [0.5, 1.5, 2.5]; % Signal timings (in min.)
34 signal_time_idx = signal_times/k; % Index of signal time in time array
35
36 % Speed breaker
37 x_spd_brk = -0.2; % Position of speed breaker
38 x_spd_brk_idx = floor((x_spd_brk - x(1))/h); % Index of speed breaker position in
   x array
39 v_spd_brk = 0.15; % Velocity at speed breaker
40
41 f = @(u) (u.^2)/2; % Flux function
42 df = @(u) u; % Derivative of Flux
43
44 % Apply boundary condition (1)
45 for i = 1:2:length(signal_time_idx)-1
46     u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx) = -1;
47     u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx + 1) = 1;
48 end
49 u(:, x_spd_brk_idx) = -1;
50
51 % Numerical calculation using Godunov method
52 for t = 1:t_steps-1
53     % Apply boundary condition (2)
```

```

54     for i = 1:2:length(signal_time_idx)-1
55         u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx) = -1;
56         u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx + 1) = 1;
57     end
58     u(:, x_spd_brk_idx) = -1;
59
60     for i = 2:x_divs - 1
61         u(t+1,i) = u(t,i) - (k/h)*(f(u_star(t,i, u,f, df))-f(u_star(t,i-1, u,f, df))
62             );
63         if i == x_spd_brk_idx
64             u(t + 1, i) = v_spd_brk * u(t + 1, i);
65         end
66     end
67     u(:,1) = u(:,2);
68     u(:,end) = u(:,end-1);
69
70     % Get rho value from u
71     rho_tab = (rhoMax / 2).*(1 - u);
72
73
74     % Plotting at different cases
75     figure;
76     % First subplot - when green light (initial cond.)
77     subplot(2, 2, 1);
78     plot(x, rho_tab(1,:), 'LineWidth', 1.5);
79     % Add labels and title
80     xlabel('Road');
81     ylabel('Traffic density');
82     title('t = 0, Uniform traffic flow (Green signal)');
83     % Display grid
84     grid on;
85     hold on;
86     % Add a vertical line at signal position
87     xline(x_signal, 'g--', 'LineWidth', 2); % Signal position
88     xline(x_spd_brk, 'm--', 'LineWidth', 1); % Speed breaker position
89     hold off;
90
91     % Second subplot - when red light
92     subplot(2, 2, 2);
93     plot(x, rho_tab(10,:), 'LineWidth', 1.5);
94     % Add labels and title
95     xlabel('Road');
96     ylabel('Traffic density');
97     title('During Red signal');
98     grid on;
99     hold on;
100    xline(x_signal, 'r--', 'LineWidth', 2); % Add a vertical line at signal position
101    xline(x_spd_brk, 'm--', 'LineWidth', 1); % Speed breaker position
102    hold off;
103
104    % Third subplot - when red light (just before green light)
105    subplot(2, 2, 3);
106    plot(x, rho_tab(15,:), 'LineWidth', 1.5);
107    % Add labels and title
108    xlabel('Road');
109    ylabel('Traffic density');
110    title('Just before Green signal');
111    grid on;

```



```

112 hold on;
113 xline(x_signal, 'r--', 'LineWidth', 2); % Add a vertical line at signal position
114 xline(x_spd_brk, 'm--', 'LineWidth', 1); % Speed breaker position
115 hold off;
116
117 % forth subplot
118 subplot(2, 2, 4);
119 plot(x, rho_tab(25,:), 'LineWidth', 1.5);
120 % Add labels and title
121 xlabel('Road');
122 ylabel('Traffic density');
123 title('Traffic flow 1 min after signal turned green');
124 grid on;
125 hold on;
126 xline(x_signal, 'g--', 'LineWidth', 2); % Add a vertical line at signal position
127 xline(x_spd_brk, 'm--', 'LineWidth', 1); % Speed breaker position
128 hold off;
129
130
131 % Function for u_star calculation used in Godunov scheme
132 function u_star_out = u_star(t, i, u, f, df)
133     gradF = df(u(t, i)); % Gradient of flux for given u(n,i)
134     gradF1 = df(u(t, i + 1)); % Gradient of flux for given u(n,i+1)
135     % Apply different conditions...
136     if gradF >= 0 && gradF1 >= 0
137         u_star_out = u(t, i);
138     elseif gradF < 0 && gradF1 < 0
139         u_star_out = u(t, i + 1);
140     elseif gradF >= 0 && gradF1 < 0
141         s = (f(u(t, i + 1)) - f(u(t, i))) / (u(t, i + 1) - u(t, i));
142         if s >= 0
143             u_star_out = u(t, i);
144         else
145             u_star_out = u(t, i + 1);
146         end
147     else
148         u_star_out = 0;
149     end
150 end

```

Output:

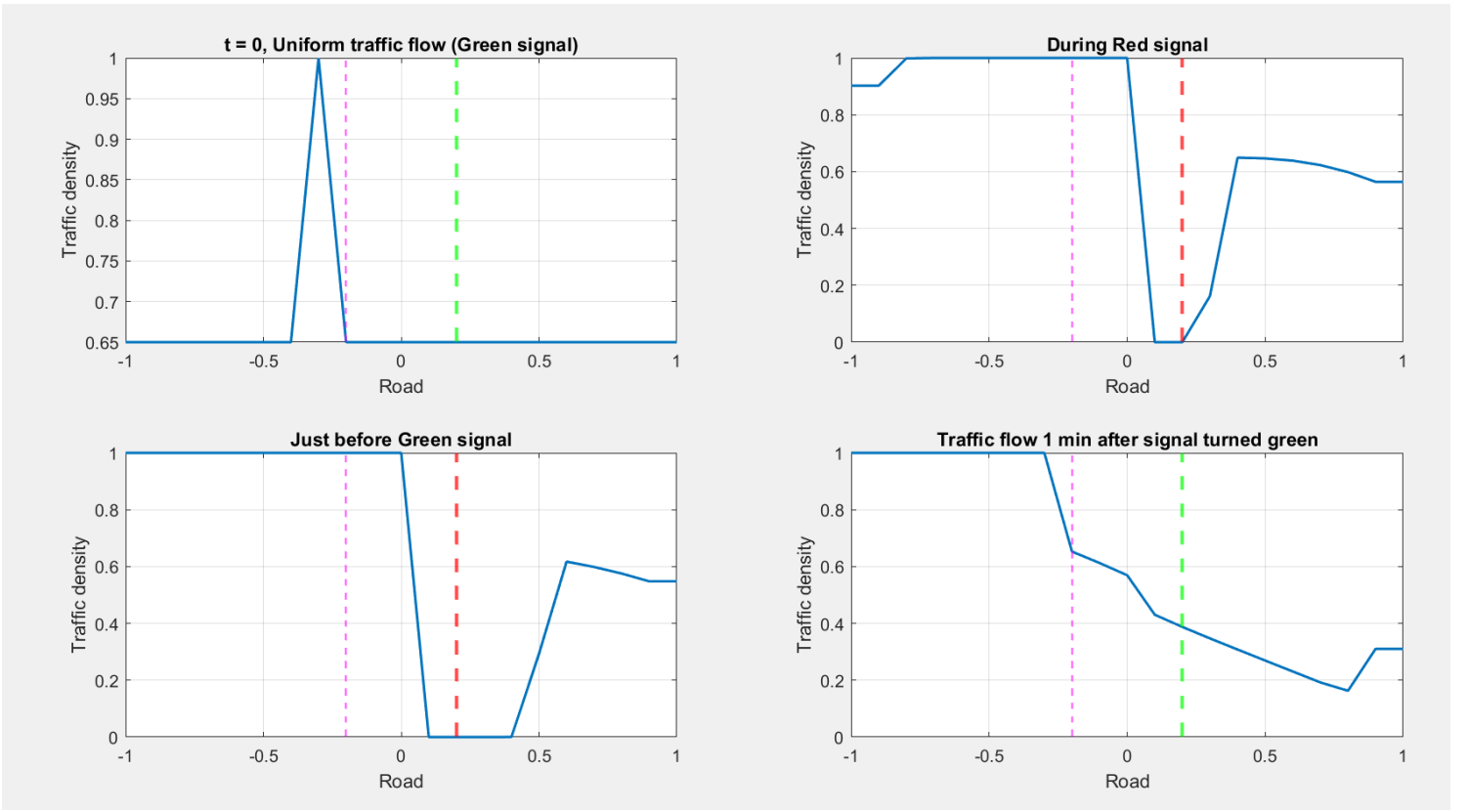


Figure 3: Density plot for different traffic signal conditions with Speed breaker before signal

		Spd brk pos.										Traffic signal position										
t/x		-1	-0.9	-0.8	-0.7	-0.6	-0.5	-0.4	-0.3	-0.2	-0.1	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Signal turned red	0	0.65	0.65	0.65	0.65	0.65	0.65	0.65	1	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65
	0.1	0.65	0.65	0.65	0.65	0.65	0.65	0.8775	1	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65
	0.2	0.65	0.65	0.65	0.65	0.65	0.65	0.770006	0.984994	1	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.6275	0.6275
	0.3	0.65	0.65	0.65	0.65	0.65	0.700403	0.932322	0.999775	1	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.643756	0.611244	0.611244
	0.4	0.65	0.65	0.65	0.667662	0.847144	0.995195	1	1	0.65	0.65	1	0	0.65	0.65	0.65	0.65	0.65	0.648166	0.635466	0.598869	0.598869
	0.5	0.65	0.65	0.65561	0.76006	0.971853	0.999977	1	1	0.65	0.8775	1	0	0.4225	0.65	0.65	0.65	0.649453	0.644564	6.27E-01	0.589094	0.589094
	0.6	0.651715	0.651715	0.699027	0.915074	0.999185	1	1	1	0.770006	0.984994	1	0	0.195	0.65	0.65	0.649836	0.648016	0.639766	6.19E-01	0.581156	0.581156
	0.7	0.668309	0.668309	0.831702	0.991973	0.999999	1	1	1	0.932322	0.999775	1	0	0.038025	0.579475	0.649951	0.649294	0.645641	0.634327	0.611217	0.57457	0.57457
	0.8	0.750007	0.750007	0.963713	0.999935	1	1	1	1	0.995195	1	1	0	0.001446	0.388539	0.649754	0.648217	0.642474	0.628653	0.604408	5.69E-01	5.69E-01
	0.9	0.902533	0.902533	0.998618	1	1	1	1	1	0.999977	1	1	0	2.09E-06	0.16241	0.649296	0.646547	0.638727	0.623002	0.598269	5.64E-01	5.64E-01
Signal turned green	1	0.98912	0.98912	0.999998	1	1	1	1	1	1	1	1	0	4.37E-12	0.026379	0.556805	0.644316	0.634611	0.61753	0.59274	0.560119	0.560119
	1.1	0.99988	0.99988	1	1	1	1	1	1	1	1	1	0	0	0.000696	0.353315	0.641609	0.630304	0.612317	0.587754	0.556505	0.556505
	1.2	1	1	1	1	1	1	1	1	1	1	1	0	0	4.84E-07	0.125527	0.637072	0.62594	0.607403	0.583246	0.553312	0.553312
	1.3	1	1	1	1	1	1	1	1	1	1	1	0	0	2.34E-13	0.015758	0.512703	0.621615	0.602797	0.579158	0.55047	0.55047
	1.4	1	1	1	1	1	1	1	1	1	1	1	0	0	0	2.48E-04	0.293002	0.617392	0.598496	0.575439	0.547923	0.547923
	1.5	1	1	1	1	1	1	1	1	1	1	0.75	0.25	0	0	6.17E-08	0.086098	0.584245	0.594486	0.572045	0.545626	0.545626
	1.6	1	1	1	1	1	1	1	1	1	0.8125	0.6875	0.3125	0.1875	0	3.77E-15	7.41E-03	0.421858	0.590748	0.568936	0.543544	0.543544
	1.7	1	1	1	1	1	1	1	1	0.847656	0.75	0.652344	0.347656	0.25	0.152344	0	5.50E-05	0.187451	0.587265	0.56608	0.541648	0.541648
	1.8	1	1	1	1	1	1	1	1	0.789291	0.710709	0.629135	0.370865	0.289291	0.210709	0.129135	3.02E-09	0.035193	0.493945	0.563448	0.539914	0.539914
	1.9	1	1	1	1	1	1	1	1	0.75	0.682986	0.612459	0.387541	0.317014	0.25	0.182986	0.112459	0.001239	0.281925	0.561015	0.538321	0.538321
	2	1	1	1	1	1	1	1	1	0.720984	0.662149	0.599812	0.400188	0.337851	0.279016	0.220984	0.162149	0.099814	0.080719	0.514927	0.536852	0.536852
	2.1	1	1	1	1	1	1	1	1	0.698443	0.645819	0.58985	0.41015	0.354181	0.301557	0.25	0.198443	0.14582	0.093666	0.340489	0.535494	0.535494
	2.2	1	1	1	1	1	1	1	1	0.680326	0.632629	0.581777	0.418223	0.367371	0.319674	0.273121	0.226879	0.180326	0.133843	0.203013	0.51005	0.51005
	2.3	1	1	1	1	1	1	1	1	0.665399	0.621726	0.575089	0.424911	0.378274	0.334601	0.292077	0.25	0.207923	0.165723	0.157143	0.421849	0.421849
	2.4	1	1	1	1	1	1	1	1	0.65286	0.612547	0.569451	0.430549	0.387453	0.34714	0.307952	0.269268	0.230732	0.192155	0.162953	0.310406	0.310406
	2.5	1	1	1	1	1	1	1	1	0.64216	0.604704	0.564627	0.435373	0.395296	0.35784	0.321468	0.285623	0.25	0.214418	0.181785	0.232751	0.232751
	2.6	1	1	1	1	1	1	1	1	0.632914	0.597918	0.560451	0.439549	0.402082	0.367086	0.333132	0.299707	0.266542	0.233475	0.201489	0.202912	0.202912
	2.7	1	1	1	1	1	1	1	1	0.624836	0.591984	0.556796	0.443204	0.408016	0.375164	0.343311	0.311979	0.280928	0.250008	0.219562	0.202064	0.202064
	2.8	1	1	1	1	1	1	1	1	0.617713	0.586749	0.553571	0.446429	0.413251	0.382287	0.352279	0.32278	0.293569	0.264511	0.235712	0.212185	0.212185
	2.9	1	1	1	1	1	1	1	0.539578	0.611382	0.582093	0.550701	0.449299	0.417907	0.388618	0.360244	0.332365	0.304776	0.277352	0.250105	0.225174	0.225174

Figure 4: Density data with speed breaker and signal

Traffic flow with speed breaker after signal:

Code: traffic_flow_model_spdbrk_right.m

```
1 % Traffic flow model with speed breaker after signal
2 clear;
3 clc;
4
5 rhoMax = 1;      % Maximum rho
6 rhoMin = 0;      % Minimum rho
7 h = 0.1;         % space step size
8 k = 0.1;         % time step size
9 x = -1:h:1;      % Road - (-1 to 1)
10 x_divs = length(x); % No. of space steps
11
12 total_time= 3;    % Total time under consideration
13 t_steps = total_time / k; % No. of time steps
14
15 % Initial conditions
16 rho0 = 0.65 * ones(1, x_divs);
17
18 % rho0 = np.linspace(0.1, 0.5, num_divs)
19 % rho0 = np.zeros(num_divs)
20 % rho0[x <= 0] = 1
21 % rho0[x > 0] = 0
22
23 u = zeros(t_steps, x_divs); % Array to store u
24
25 % Initial and boundary conditions for u
26 u(1,:) = 1 - 2 .* rho0/rhoMax ;
27 u(:,1) = u(1,1)*ones(t_steps,1);
28
29 % Signal
30 x_signal = 0.2; % Position of signal
31 x_signal_idx = floor((x_signal - x(1))/h); % Index of signal position in x array
32
33 signal_times = [0.5, 1.5, 2.5]; % Signal timings (in min.)
34 signal_time_idx = signal_times/k; % Index of signal time in time array
35
36 % Speed breaker
37 x_spd_brk = 0.7; % Position of speed breaker
38 x_spd_brk_idx = floor((x_spd_brk - x(1))/h); % Index of speed breaker position in
    x array
39 v_spd_brk = 0.15; % Velocity at speed breaker
40
41 f = @(u) (u.^2)/2; % Flux function
42 df = @(u) u; % Derivative of Flux
43
44 % Apply boundary condition (1)
45 for i = 1:2:length(signal_time_idx)-1
46     u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx) = -1;
47     u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx + 1) = 1;
48 end
49 u(:, x_spd_brk_idx+1) = 1;
50
51 % Numerical calculation using Godunov method
52 for t = 1:t_steps-1
```

```

53 % Apply boundary condition (2)
54 for i = 1:2:length(signal_time_idx)-1
55     u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx) = -1;
56     u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx + 1) = 1;
57 end
58 u(:, x_spd_brk_idx+1) = 1;
59
60 for i = 2:x_divs - 1
61     u(t+1,i) = u(t,i) - (k/h)*(f(u_star(t,i, u,f, df))-f(u_star(t,i-1, u,f, df))
62         );
63     if i == x_spd_brk_idx
64         u(t + 1, i) = v_spd_brk * u(t + 1, i);
65     end
66 end
67 u(:,1) = u(:,2);
68 u(:,end) = u(:,end-1);
69
70 % Get rho value from u
71 rho_tab = (rhoMax / 2).*(1 - u);
72
73
74 % Plotting at different cases
75 figure;
76 % First subplot - when green light (initial cond.)
77 subplot(2, 2, 1);
78 plot(x, rho_tab(1,:), 'LineWidth', 1.5);
79 % Add labels and title
80 xlabel('Road');
81 ylabel('Traffic density');
82 title('t = 0, Uniform traffic flow (Green signal)');
83 % Display grid
84 grid on;
85 hold on;
86 % Add a vertical line at signal position
87 xline(x_signal, 'g--', 'LineWidth', 2); % Signal position
88 xline(x_spd_brk, 'm--', 'LineWidth', 1); % Speed breaker position
89 hold off;
90
91 % Second subplot - when red light
92 subplot(2, 2, 2);
93 plot(x, rho_tab(10,:), 'LineWidth', 1.5);
94 % Add labels and title
95 xlabel('Road');
96 ylabel('Traffic density');
97 title('During Red signal');
98 grid on;
99 hold on;
100 xline(x_signal, 'r--', 'LineWidth', 2); % Add a vertical line at signal position
101 xline(x_spd_brk, 'm--', 'LineWidth', 1); % Speed breaker position
102 hold off;
103
104 % Third subplot - when red light (just before green light)
105 subplot(2, 2, 3);
106 plot(x, rho_tab(15,:), 'LineWidth', 1.5);
107 % Add labels and title
108 xlabel('Road');
109 ylabel('Traffic density');
110 title('Just before Green signal');

```

```

111 grid on;
112 hold on;
113 xline(x_signal, 'r--', 'LineWidth', 2); % Add a vertical line at signal position
114 xline(x_spd_brk, 'm--', 'LineWidth', 1); % Speed breaker position
115 hold off;
116
117 % forth subplot
118 subplot(2, 2, 4);
119 plot(x, rho_tab(25,:), 'LineWidth', 1.5);
120 % Add labels and title
121 xlabel('Road');
122 ylabel('Traffic density');
123 title('Traffic flow 1 min after signal turned green');
124 grid on;
125 hold on;
126 xline(x_signal, 'g--', 'LineWidth', 2); % Add a vertical line at signal position
127 xline(x_spd_brk, 'm--', 'LineWidth', 1); % Speed breaker position
128 hold off;
129
130
131 % Function for u_star calculation used in Godunov scheme
132 function u_star_out = u_star(t, i, u, f, df)
133     gradF = df(u(t, i)); % Gradient of flux for given u(n,i)
134     gradF1 = df(u(t, i + 1)); % Gradient of flux for given u(n,i+1)
135     % Apply different conditions...
136     if gradF >= 0 && gradF1 >= 0
137         u_star_out = u(t, i);
138     elseif gradF < 0 && gradF1 < 0
139         u_star_out = u(t, i + 1);
140     elseif gradF >= 0 && gradF1 < 0
141         s = (f(u(t, i + 1)) - f(u(t, i))) / (u(t, i + 1) - u(t, i));
142         if s >= 0
143             u_star_out = u(t, i);
144         else
145             u_star_out = u(t, i + 1);
146         end
147     else
148         u_star_out = 0;
149     end
150 end

```

Output:

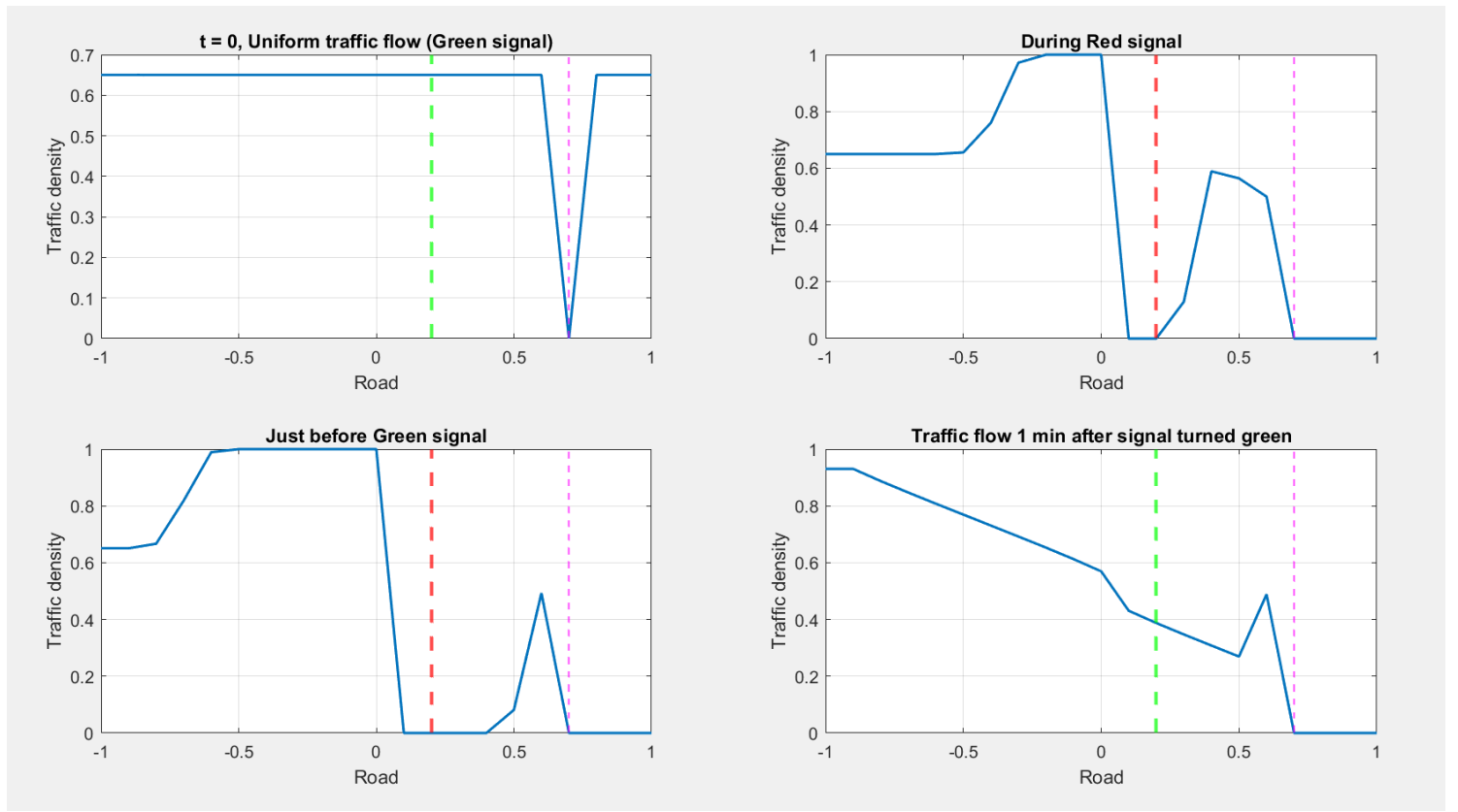


Figure 5: Density plot for different traffic signal conditions with Speed breaker before signal

		Traffic signal position												Speed breaker pos											
t/x		-1	-0.9	-0.8	-0.7	-0.6	-0.5	-0.4	-0.3	-0.2	-0.1	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1			
Signal turned red	0	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0	0.65	0.65	0.65			
	0.1	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.519125	0	0.4225	0.65	0.65			
	0.2	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.627866	0.502814	0	0.195	0.6275	0.6275		
	0.3	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.64385	0.611524	0.500421	0	0.038025	0.534475	0.534475	
	0.4	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.648193	0.635595	0.599087	0.500063	0	0.001446	0.321054	0.321054
	0.5	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.8775	1	0	0.421961	0.644618	0.627027	0.589268	0.500009	0	2.09E-06	0.10452	0.10452		
	0.6	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.770006	0.984994	1	0	0	0.192875	0.639839	0.61886	0.5813	0.500001	0	4.37E-12	0.010926	0.010926		
	0.7	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.700403	0.932322	0.999775	1	0	0.037201	0.559641	0.611342	0.57469	0.5	0	0.000119	0.000119			
	0.8	0.65	0.65	0.65	0.65	0.65	0.65	0.65	0.667662	0.847144	0.955195	1	1	0	0.001384	0.357855	0.604523	0.569111	0.5	0	1.43E-08	1.43E-08			
	0.9	0.65	0.65	0.65	0.65	0.65	0.65561	0.76006	0.971853	0.999977	1	1	0	0	1.92E-06	0.129442	0.589095	0.564335	0.5	0	2.22E-16	2.22E-16			
Signal turned green	1	0.65	0.65	0.65	0.65	0.651715	0.699027	0.915074	0.999185	1	1	1	0	0	3.67E-12	0.016575	0.455921	0.560196	0.5	0	0	0			
	1.1	0.65	0.65	0.65	0.650517	0.668309	0.831702	0.991973	0.999999	1	1	1	0	0	0.000281	0.226021	0.556572	0.5	0	0	0				
	1.2	0.65	0.65	0.650155	0.65619	0.750007	0.963713	0.999935	1	1	1	1	0	0	7.89E-08	0.051366	0.481508	0.5	0	0	0				
	1.3	0.650047	0.650047	0.652004	0.694298	0.902533	0.998618	1	1	1	1	1	0	0	6.22E-15	0.002639	0.280577	0.499949	0.5	0	0	0			
	1.4	0.650638	0.650638	0.666651	0.818579	0.98912	0.999998	1	1	1	1	1	0	0	0	6.96E-06	0.081355	0.49277	0.5	0	0	0			
	1.5	0.655719	0.655719	0.740371	0.956325	0.99988	1	1	1	1	1	0.75	0.25	0	0	4.85E-11	0.006626	0.472634	0.5	0	0	0			
	1.6	0.689248	0.689248	0.890825	0.997972	1	1	1	1	1	1	0.8125	0.6875	0.3125	0.1875	0	0.00E+00	4.39E-05	0.459495	0	0	0	0		
	1.7	0.806178	0.806178	0.986057	0.999996	1	1	1	1	1	0.847656	0.75	0.652344	0.347656	0.25	0.152344	0	1.93E-09	0.456677	0	0	0	0		
	1.8	0.948685	0.948685	0.999801	1	1	1	1	1	0.870865	0.789291	0.710709	0.629135	0.370865	0.289291	0.210709	0.129135	0.00E+00	0.456283	0	0	0	0		
	1.9	0.997168	0.997168	1	1	1	1	1	0.887541	0.817014	0.75	0.682986	0.612459	0.387541	0.317014	0.25	0.182986	0.112459	0.456229	0	0	0	0		
2	0.999992	0.999992	1	1	1	0.900188	0.837851	0.779016	0.720984	0.662149	0.599812	0.400188	0.337851	0.279016	0.220984	0.162149	0.471194	0.471194	0	0	0	0			
2.1	1	1	1	1	0.91015	0.854181	0.801557	0.75	0.698443	0.645819	0.58985	0.41015	0.354181	0.301557	0.25	0.198443	0.479682	0.479682	0	0	0	0			
2.2	1	1	1	0.918223	0.867371	0.819674	0.773121	0.726879	0.680326	0.632629	0.581777	0.418223	0.367371	0.319674	0.273121	0.226879	0.48323	0.48323	0	0	0	0			
2.3	1	1	0.924911	0.878274	0.834601	0.792077	0.75	0.70923	0.665399	0.621726	0.575089	0.424911	0.378274	0.334601	0.292077	0.25	0.486337	0.486337	0	0	0	0			
2.4	0.930549	0.930549	0.887453	0.84714	0.807952	0.769268	0.730732	0.692048	0.65286	0.612547	0.569451	0.430549	0.387453	0.34714	0.307952	0.269268	0.488604	0.488604	0	0	0	0			
2.5	0.895296	0.895296	0.85784	0.821468	0.785623	0.75	0.714377	0.678532	0.64216	0.604704	0.564627	0.435373	0.395296	0.35784	0.321468	0.285623	0.490324	0.490324	0	0	0	0			
2.6	0.867086	0.867086	0.833132	0.799707	0.766542	0.733458	0.700293	0.666686	0.632914	0.599718	0.560451	0.439549	0.402082	0.367086	0.333132	0.299707	0.491669	0.491669	0	0	0	0			
2.7	0.843311	0.843311	0.811979	0.780928	0.75	0.719072	0.688021	0.656689	0.624836	0.591984	0.556796	0.443204	0.408016	0.375164	0.343311	0.311979	0.492743	0.492743	0	0	0	0			
2.8	0.82278	0.82278	0.793569	0.764507	0.735493	0.706431	0.67722	0.647721	0.617713	0.586749	0.555371	0.446429	0.413251	0.382287	0.352279	0.32278	0.493617	0.493617	0	0	0	0			
2.9	0.804776	0.804776	0.77735	0.75	0.72265	0.695224	0.667635	0.639756	0.611382	0.582093	0.550701	0.449299	0.417907	0.388618	0.360244	0.332635	0.494338	0.494338	0.249595	0	0	0			

Figure 6: Enter Caption

- Implement Gudanov Method to numerically solve the same PDE equation when the speed density relation follows the car following model, that is the traffic microscopic model. Generate the same tables as in Question no 1 for each case. This creates a bridge between traffic macroscopic and microscopic model.

```

1 % Traffic flow model for speed-density relation follows the car following model.
2 clear;
3 clc;
4
5 rhoMax = 1;      % Maximum rho
6 rhoMin = 0;      % Minimum rho
7
8 h = 0.1;         % space step size
9 k = 0.1;         % time step size
10 x = -1:h:1;      % Road - (-1 to 1)
11 x_divs = length(x); % No. of space steps
12
13 total_time= 3;    % Total time under consideration
14 t_steps = total_time / k; % No. of time steps
15
16 % Initial conditions
17 rho0 = linspace(0.1, 0.5, x_divs);
18
19 u = zeros(t_steps, x_divs); % Array to store u
20
21 % Initial and boundary conditions for u
22 u(1,:) = rho0; % 1 - 2 .* rho0/rhoMax ;
23 u(:,1) = u(1,1)*ones(t_steps,1);
24
25 % Numerical calculation using Godunov method
26 for t = 1:t_steps-1
27     % Apply boundary condition (2)
28     % for i = 1:2:length(signal_time_idx)-1
29     %     u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx) = -1;
30     %     u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx + 1) = 1;
31     % end
32
33     for i = 2:x_divs - 1
34         u(t+1,i) = u(t,i) - (k/h)*(f(u_star(t,i, u,@f, @df))-f(u_star(t,i-1, u,@f, @
35         df)));
36     end
37 end
38 u(:,1) = u(:,2);
39 u(:,end) = u(:,end-1);
40
41 % Get rho value from u
42 rho_tab = u;
43
44 % Plotting at different cases
45 figure;
46 % First subplot
47 subplot(2, 2, 1);
48 plot(x, rho_tab(1,:), 'LineWidth', 1.5);
49 % Add labels and title
50 xlabel('Road');
51 ylabel('Traffic density');
52 title('t = 0');

```

```

53 % Display grid
54 grid on;
55
56
57 % Second subplot
58 subplot(2, 2, 2);
59 plot(x, rho_tab(10,:), 'LineWidth', 1.5);
60 % Add labels and title
61 xlabel('Road');
62 ylabel('Traffic density');
63 title('t1');
64 grid on;
65
66 % Third subplot
67 subplot(2, 2, 3);
68 plot(x, rho_tab(15,:), 'LineWidth', 1.5);
69 % Add labels and title
70 xlabel('Road');
71 ylabel('Traffic density');
72 title('t2');
73 grid on;
74
75 % forth subplot
76 subplot(2, 2, 4);
77 plot(x, rho_tab(25,:), 'LineWidth', 1.5);
78 % Add labels and title
79 xlabel('Road');
80 ylabel('Traffic density');
81 title('t3');
82 grid on;
83
84 % Function for u_star calculation used in Godunov scheme
85 function u_star_out = u_star(n, i, u, f, df)
86     gradF = df(u(n, i)); % Gradient of flux for given u(n,i)
87     gradF1 = df(u(n, i + 1)); % Gradient of flux for given u(n,i+1)
88     % Apply different conditions...
89     if gradF >= 0 && gradF1 >= 0
90         u_star_out = u(n, i);
91     elseif gradF < 0 && gradF1 < 0
92         u_star_out = u(n, i + 1);
93     elseif gradF >= 0 && gradF1 < 0
94         s = (f(u(n, i + 1)) - f(u(n, i))) / (u(n, i + 1) - u(n, i));
95         if s >= 0
96             u_star_out = u(n, i);
97         else
98             u_star_out = u(n, i + 1);
99         end
100     else
101         u_star_out = 0;
102     end
103 end
104
105 % Flux function for car following model
106 function result = f(rho)
107     rhoCrit = 0.45; % Critical rho
108     rhoJam = 1; % Jam rho
109     vMax = 0.95; % Max velocity
110
111     if rho <= rhoCrit

```



```

112     result = rho * vMax;
113 elseif rhoCrit < rho && rho < rhoJam
114     result = vMax * (1 - rho / rhoJam);
115 elseif rho >= rhoJam
116     result = 0;
117 end
118 end
119
120 % Derivative of flux
121 function result = df(rho)
122     rhoCrit = 0.45; % Critical rho
123     rhoJam = 1;     % Jam rho
124     vMax = 0.95;    % Max velocity
125
126     if rho <= rhoCrit
127         result = vMax;
128     elseif rhoCrit < rho && rho < rhoJam
129         result = -vMax / rhoJam;
130     elseif rho >= rhoJam
131         result = 0;
132     end
133 end

```

Output:

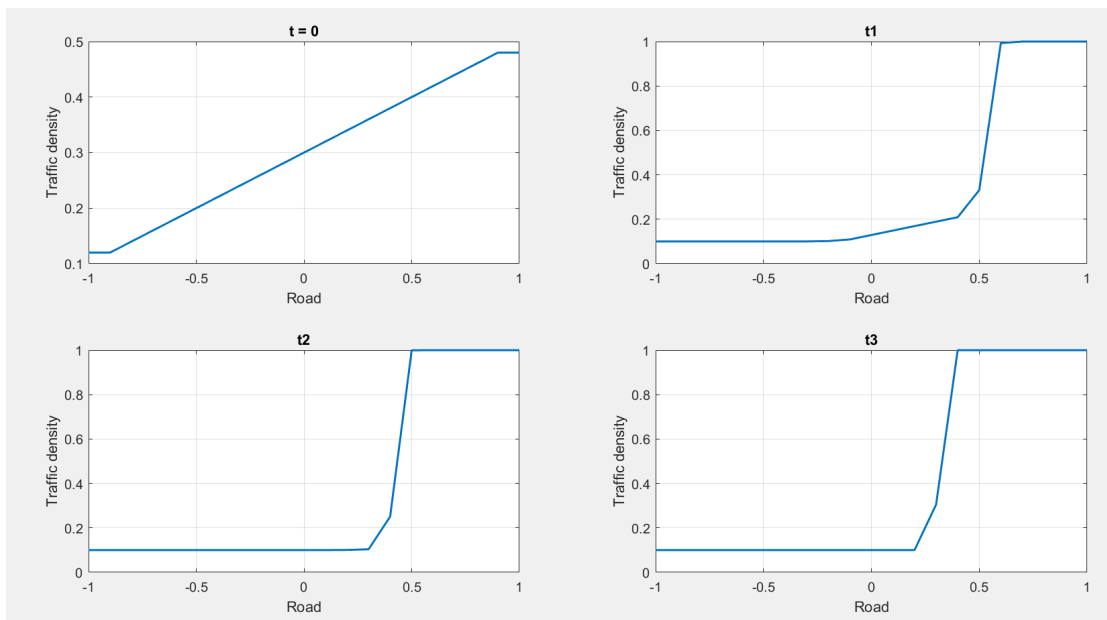


Figure 7: Density plots for different t for car following model

Car following model with signal

```

1  % Traffic flow model for speed-density relation follows the car following model with
    signal
2  clear;
3  clc;
4
5  rhoMax = 1;      % Maximum rho
6  rhoMin = 0;      % Minimum rho
7
8  h = 0.1;         % space step size
9  k = 0.1;         % time step size
10 x = -1:h:1;      % Road - (-1 to 1)
11 x_divs = length(x);      % No. of space steps
12
13 total_time= 3;    % Total time under consideration
14 t_steps = total_time / k;      % No. of time steps
15
16 % Initial conditions
17 rho0 = linspace(0.1, 0.5, x_divs);
18
19 u = zeros(t_steps, x_divs);    % Array to store u
20
21 % Initial and boundary conditions for u
22 u(1,:) = rho0;
23 u(:,1) = u(1,1)*ones(t_steps,1);
24
25 % Signal
26 x_signal = 0.2;    % Position of signal
27 x_signal_idx = floor((x_signal - x(1))/h); % Index of signal position in x array
28
29 signal_times = [0.5, 1.5, 2.5]; % Signal timings (in min.)
30 signal_time_idx = signal_times/k; % Index of signal time in time array
31
32 % Apply boundary condition (1)
33 for i = 1:2:length(signal_time_idx)-1
34     u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx) = -1;
35     u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx + 1) = 1;
36 end
37
38 % Numerical calculation using Godunov method
39 for t = 1:t_steps-1
40     % Apply boundary condition (2)
41     for i = 1:2:length(signal_time_idx)-1
42         u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx) = -1;
43         u(signal_time_idx(i):signal_time_idx(i+1), x_signal_idx + 1) = 1;
44     end
45
46     for i = 2:x_divs - 1
47         u(t+1,i) = u(t,i) - (k/h)*(f(u_star(t,i, u,@f, @df))-f(u_star(t,i-1, u,@f, @
            df)));
48     end
49 end
50 u(:,1) = u(:,2);
51 u(:,end) = u(:,end-1);
52
53 % Get rho value from u
54 rho_tab = u;

```

```

55
56
57 % Plotting at different cases
58 figure;
59 % First subplot - when green light (initial cond.)
60 subplot(2, 2, 1);
61 plot(x, rho_tab(1,:), 'LineWidth', 1.5);
62 % Add labels and title
63 xlabel('Road');
64 ylabel('Traffic density');
65 title('t = 0, Uniform traffic flow (Green signal)');
66 % Display grid
67 grid on;
68 hold on;
69 % Add a vertical line at signal position
70 xline(x_signal, 'g--', 'LineWidth', 2); % Signal position
71 hold off;
72
73 % Second subplot - when red light
74 subplot(2, 2, 2);
75 plot(x, rho_tab(10,:), 'LineWidth', 1.5);
76 % Add labels and title
77 xlabel('Road');
78 ylabel('Traffic density');
79 title('During Red signal');
80 grid on;
81 hold on;
82 % Add a vertical line at signal position
83 xline(x_signal, 'r--', 'LineWidth', 2);
84 hold off;
85
86 % Third subplot - when red light (just before green light)
87 subplot(2, 2, 3);
88 plot(x, rho_tab(15,:), 'LineWidth', 1.5);
89 % Add labels and title
90 xlabel('Road');
91 ylabel('Traffic density');
92 title('Just before Green signal');
93 grid on;
94 hold on;
95 % Add a vertical line at signal position
96 xline(x_signal, 'r--', 'LineWidth', 2);
97 hold off;
98
99 % forth subplot
100 subplot(2, 2, 4);
101 plot(x, rho_tab(25,:), 'LineWidth', 1.5);
102 % Add labels and title
103 xlabel('Road');
104 ylabel('Traffic density');
105 title('Traffic flow 1 min after signal turned green');
106 grid on;
107 hold on;
108 % Add a vertical line at signal position
109 xline(x_signal, 'g--', 'LineWidth', 2);
110 hold off;
111
112
113

```

```

114 % Function for u_star calculation used in Godunov scheme
115 function u_star_out = u_star(n, i, u, f, df)
116     gradF = df(u(n, i)); % Gradient of flux for given u(n,i)
117     gradF1 = df(u(n, i + 1)); % Gradient of flux for given u(n,i+1)
118     % Apply different conditions...
119     if gradF >= 0 && gradF1 >= 0
120         u_star_out = u(n, i);
121     elseif gradF < 0 && gradF1 < 0
122         u_star_out = u(n, i + 1);
123     elseif gradF >= 0 && gradF1 < 0
124         s = (f(u(n, i + 1)) - f(u(n, i))) / (u(n, i + 1) - u(n, i));
125         if s >= 0
126             u_star_out = u(n, i);
127         else
128             u_star_out = u(n, i + 1);
129         end
130     else
131         u_star_out = 0;
132     end
133 end
134
135 % Flux function for car following model
136 function result = f(rho)
137     rhoCrit = 0.45; % Critical rho
138     rhoJam = 1; % Jam rho
139     vMax = 0.95; % Max velocity
140
141     if rho <= rhoCrit
142         result = rho * vMax;
143     elseif rhoCrit < rho && rho < rhoJam
144         result = vMax * (1 - rho / rhoJam);
145     elseif rho >= rhoJam
146         result = 0;
147     end
148 end
149
150 % Derivative of flux
151 function result = df(rho)
152     rhoCrit = 0.45; % Critical rho
153     rhoJam = 1; % Jam rho
154     vMax = 0.95; % Max velocity
155
156     if rho <= rhoCrit
157         result = vMax;
158     elseif rhoCrit < rho && rho < rhoJam
159         result = -vMax / rhoJam;
160     elseif rho >= rhoJam
161         result = 0;
162     end
163 end

```

Output:

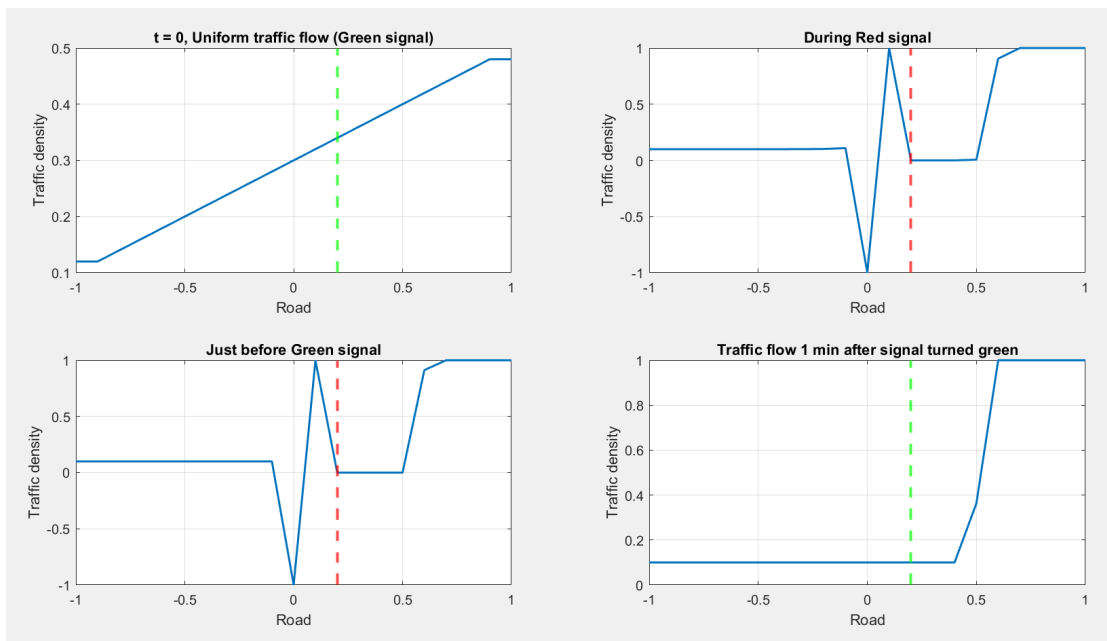


Figure 8: Density plots for different t for car following model with traffic signal

3. How will you formulate the traffic flow modelling problem as a black box model?

To formulate the traffic flow model to a black box model, we focus on observing the inputs and collecting the outputs of the system instead of looking at the complex formulations. Some of the input variables for the black box model are:

- (a) Traffic Density - vehicles per unit length
- (b) Traffic Speed - avg. speed of vehicle on the road
- (c) Traffic Flow - no. of vehicles passing through a point per unit time.
- (d) Traffic Composition - types of vehicles on the road.
- (e) External variables: Road type, Time of day, Weather, etc.

We collect data for these inputs and train the black-box model to predict the traffic flow of the road and conditions like traffic jams etc.
