UC Berkeley Math 228B, Spring 2022: Problem Set 6

Prof. Per-Olof Persson (persson@berkeley.edu)

Due April 22

Consider the traffic flow problem, described by the non-linear hyperbolic equation:

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} = 0 \tag{1}$$

with $\rho = \rho(x,t)$ the density of cars (vehicles/km), and $u = u(x,t)$ the velocity. Assume that the velocity u is given as a function of ρ:

$$u = u_{\max}\left(1 - \frac{\rho}{\rho_{\max}}\right), \tag{2}$$

with $u_{\max}$ the maximum speed and $0 \le \rho \le \rho_{\max}$. The flux of cars is therefore given by:

$$f(\rho) = \rho u_{\max}\left(1 - \frac{\rho}{\rho_{\max}}\right). \tag{3}$$

We will solve this problem using a first order finite volume scheme:

$$\rho_i^{n+1} = \rho_i^n - \frac{\Delta t}{\Delta x}\left(F^n_{i+\frac{1}{2}} - F^n_{i-\frac{1}{2}}\right). \tag{4}$$

For the numerical flux function, we will consider two different schemes:

Roe's Scheme

The expression of the numerical flux is given by:

$$F^R_{i+\frac{1}{2}} = \frac{1}{2}\left[f(\rho_i) + f(\rho_{i+1})\right] - \frac{1}{2}\left|a_{i+\frac{1}{2}}\right|(\rho_{i+1} - \rho_i) \tag{5}$$

with

$$a_{i+\frac{1}{2}} = u_{\max}\left(1 - \frac{\rho_i + \rho_{i+1}}{\rho_{\max}}\right). \tag{6}$$

Note that $a_{i+\frac{1}{2}}$ satisfies

$$f(\rho_{i+1}) - f(\rho_i) = a_{i+\frac{1}{2}}(\rho_{i+1} - \rho_i). \tag{7}$$

Godunov's Scheme

In this case the numerical flux is given by:

$$F^G_{i+\frac{1}{2}} = f\left(\rho\left(x_{i+\frac{1}{2}}, t^{n+}\right)\right) = \begin{cases} \min_{\rho\in[\rho_i,\rho_{i+1}]} f(\rho), & \rho_i < \rho_{i+1} \\ \max_{\rho\in[\rho_i,\rho_{i+1}]} f(\rho), & \rho_i > \rho_{i+1}. \end{cases} \tag{8}$$

1. For both Roe's Scheme and Godunov's Scheme, look at the problem of a traffic light turning green at time $t = 0$. We are interested in the solution at $t = 2$ using both schemes. Use the following problem parameters:

$$u_{\max} = 1.0, \quad \rho_{\max} = 1.0, \quad \rho_L = 0.8 \tag{9}$$

Solve on the domain $x \in [-2, 2]$ using 401 cells, that is, the cell midpoints are located at $x_j = -2 + j\Delta x$ for $j = 0, \ldots, N$, with $N = 400$ and $\Delta x = 4/N$. Note that the cell centers are located between the cell centers, so e.g. the left endpoint of the middle cell is located at $x_{-1/2} = -\Delta x/2$. To satisfy the CFL condition, use a timestep of $\Delta t = \frac{0.8\Delta x}{u_{\max}}$.

The initial condition at the instant when the traffic light turns green is

$$\rho(x, t = 0) = \begin{cases} \rho_L, & x < 0 \\ 0, & x \geq 0 \end{cases} \tag{10}$$

Finally, you need to specify boundary conditions, that is, the numerical fluxes at the endpoints of the domain. For simplicity, simply freeze ρ in the first and the last cells, which means no boundary fluxes are needed (this will implement a so-called characteristic boundary condition, where your numerical flux function will decide if the boundary is an inflow or an outflow).

What do you observe for each of the schemes? Explain briefly why the behavior you get arises.
For problems 2 - 3, use only the scheme(s) which are valid (that is, converging to the weak entropy solution).

2. Simulate the effect of a traffic light at $x = -\frac{\Delta x}{2}$ which has a period of $T = T_1 + T_2 = 2$ time units. Assume that the traffic light is $T_1 = 1$ units on red and $T_2 = 1$ units on green. Set $\rho_L = \frac{\rho_{\max}}{2}$ on the left boundary, giving a maximum flux and a high density of incoming cars. Determine the average flow, or the *capacity* of cars over a time period T.

The average flow can be approximated as

$$\dot{q} = \frac{1}{N_T} \sum_{n=1}^{N_T} f^n = \frac{1}{N^T} \sum_{n=1}^{N_T} \rho^n u^n, \tag{11}$$

where superscripts indicate the timestep and N_T is the number of time steps for each period T. You should run your computation until $\dot{q}$ over a time period does not change. Note that because of continuity, this can be evaluated at any cell of the interior of the domain (not at the boundaries since they are frozen).

Note: A red traffic light can be modeled by simply setting $F_{i-1/2} = 0$ at the position where the traffic light is located.

3. Assume now that we simulate two traffic lights, one located at $x = 0$, and the other at $x = 0.15$, both with a period T. Calculate the road capacity (= average flow) for different delay factors. That is if the first light turns green at time t, then the second light will turn green at $t + \tau$. Solve for $\tau = k\frac{T}{10}$, $k = 0, \ldots, 9$. Plot your results of capacity vs τ and determine the optimal delay τ.

4. ***Optional, not graded***. Consider the Eikonal equation for first arrivals/optimal path planning problems. The equation can be written

$$F(x,y)|\nabla\phi(x,y)| = 1,$$

where $F(x,y)$ is the speed function. We will use the solution $\phi(x,y)$ to determine the optimal path from the departure point (x_D, y_D) to the arrival point (x_A, y_A), where $\phi(x_A, y_A) = t$. The level set with value t of the function $\phi(x,y)$ gives the maximum distance away from our departure point that can be traveled in a time t. In addition, the optimal path between the departure point and the arrival point is determined by traveling in the normal direction of the level sets.

We will solve the Eikonal equation using a level set method and a time-stepping approach. The equation

$$\phi_t + F|\nabla\phi| = 1, \qquad \phi(x_D, y_D) = 0$$

is integrated in time until a steady-state is reached. This is not an efficient method for solving the Eikonal equation, but it will be sufficient for this problem set (and it illustrates how to solve more general time-dependent problems). If you are interested in more sophisticated solvers, feel free to implement the more efficient fast marching method instead.

(a) Write a computer code to solve the Eikonal equation on the unit square $x, y \in [0,1]$. Use the first-order upwinded scheme in space, and appropriate treatment of the boundaries.

(b) Write a computer code to find the optimal path between the departure/arrival point, by solving the ODE $d\boldsymbol{r}/dt = \boldsymbol{n}$, where $\boldsymbol{r}$ is the current position on the path and $\boldsymbol{n}$ is the normal vector.

(c) Run your codes with grid spacing $h = 1/100$ for the following cases and plot both the solutions (e.g. as contour curves of $\phi(x,y)$) and the optimal paths:

Case 1: $(x_D, y_D) = (0.2, 0.2)$, $(x_A, y_A) = (0.8, 0.8)$, $F(x,y) = 1$ (for testing).
Case 2: $(x_D, y_D) = (0.2, 0.2)$, $(x_A, y_A) = (0.8, 0.45)$, and

$$F(x,y) = \begin{cases} 1.0 & \text{if } y \geq 0.5, \\ 0.5 & \text{if } y < 0.5. \end{cases}$$

Case 3: $(x_D, y_D) = (0.2, 0.2)$, $(x_A, y_A) = (0.8, 0.8)$, and

$$F(x,y) = 1 - 0.9 \cdot \cos(4\pi x) \cdot e^{-10\,((x-.5)^2 + (y-0.5)^2)}$$

Case 4: Make up your own speed function F, only returning the values 0.01 and 1 but with a non-trivial optimal path.