Advection with Upwinding and BFECC

This project is due at 11 PM on Tuesday, April 5.

In this project you will write two codes to approximate solutions of

$$\begin{aligned} \partial_t u(x,t) + v(x)\partial_x u(x,t) &= 0 \text{ on } [0,1]\times[0,T], \\ u(0,t) &= u(1,t) \text{ on } [0,T], \\ u(x,0) &= g(x) \text{ on } [0,1], \end{aligned}$$

where v, T, and g are given.

Your codes should compute numbers U_j^m that approximate the value of $u(j\,dx, m\,dt)$, where $dx = 1/N$, dt is given. Here N is a positive integer. It will be the case that dt will be chosen so that $dt \max_{x\in[0,1]} |v| \le dx$.

I suggest that your code should maintain *time* as a variable and store the solution corresponding to that time in a vector U of length $N+1$, with $U_0 = U_N$. At appropriate values of *time* you should save or write the values in U so that you can make the plots requested.

First Order Upwind

Given the approximation solution U at some time t, we want to compute W, the approximate solution corresponding to $t + dt$. The scheme to us is given by the pseudo code here.

```
for j = 0, ..., N-1
   frac = v( j dx) * dt / dx
   if frac < 0
      k = j + 1
      frac = -frac
   else
      k = j - 1 mod N
   W[j] = U[j] + frac*(U[k]-U[j])
W[N] = W[0]
```

(Note that any integer mod N should be in $[0, N-1]$.)

Suppose that this is encapsulated as a function `step_upwind( N, dt, dx, v, U )` that returns W.

Case 0

Test your code with $v \equiv 1$, $T = 1$, $N = 80$, $dt = dx = 1/N$. The initial condition

$$g(x) = \begin{cases} 20x \text{ on } [0, 0.05] \\ 2 - 20x \text{ on } [0.05, 0.1] \\ 0 \text{ otherwise .} \end{cases}$$

Produce a single graph that shows `U` at $t = 0$, $t = 1/4$, $t = 1/2$, and $t = 1$.

Case 1

Next use the same parameters with $v \equiv -1$ and produce a corresponding graph.

The results of these experiments should be essentially perfect. If they are not, it indicates a bug.

Case 2

Next repeat Case 0, except using $dt = dx/2$.

Back & Forth Error Compensation & Correction (BFECC)

In this scheme, to advance U, the approximate solution at t to $t + dt$ we us `step_upwind` three time as follows:

```
G = setp_upwind(N, dt, dx, v, U)
B = setp_upwind(N, -dt, dx, v, G)
```

```
C = U + (1/2) (U-B)
W = setp_upwind(N, dt, dx, v, C)
```

For the PDE `B` would be the same as `U`, but `U` was smoothed when moving to `G` and again when moving back to `B`. In `C` we added in the amount we expect `U` to loose when moved forward with `step_upwind`.

Case 3

Redo Case 2 using BFECC.

Case 4

Redo Case 3 with $N = 320$, $dx = 1/N$, and $dt = dx/2$.

Case 5

Use the same initial condition, $g()$, as above. Take $N = 320$, $dt = dx/2$, and take

$$v(x) = \begin{cases} 1 \text{ on } \text{ on } [0, 1/4] \\ 1 - 2(x - 1/4) \text{ on } (1/4, 1/2] \\ 1/2 \text{ on } (1/2, 3/4] \\ 1/2 + 2(x - 3/4) \text{ on } (3/4, 1]. \end{cases}$$

Stop when $t > 3/4 + ln(2)$. So we have something to call it, lets refer to this as a ramped velocity.

Plot the solution on a single plot at about $t = 0.3 * k$ for $k = 0, 1, 2, \ldots$.

For each plot that you make there should be a title or a caption that gives some information about the parameters and scheme used in computing the solutions. For example Case 3 might have a title or caption

> Snapsots produce by BFECC using N=80, dt=dx/2, v=1, at times t = 0, 1/4, 1/2, 1.

Last revised 3/30/22