Problem Set 3

(Out Thu 09/22/2022, Due Tue 10/04/2022)

Problem 3

Consider the same 1d advection-reaction equation as in problem set 2:

$$\phi_t + u\phi_x = g(\phi)$$

on the domain $x \in [-1, 1]$, where the flow velocity field is $u(x) = \sin(2\pi x)$, the reaction term is $g(\phi) = -6(\phi - 1)\phi(\phi + 1)$, and the initial state is $\phi(x, 0) = \sin(\pi x)$. We are interested in the solution at $t = 3$.

(a) Derive and program a first-order semi-Lagrangian scheme (using backwards tracking) to solve this problem. Use simple Euler steps for the characteristic ODEs, and simple piecewise linear interpolation (you can use Matlab's `interp1` function with the option `'linear'`). Explain in your paper submission how the scheme works, and email your code under the file name `yourfamilyname_problem3a.m` . Choose the grid spacing Δx and time step Δt so that the numerical approximation is correct in the "eye norm", i.e., it deviates from the true solution by about 1%.

(b) Modify your code to yield a third-order semi-Lagrangian scheme. To that end, replace the Euler steps by RK3 steps, and change the piecewise linear interpolation to a piecewise cubic (e.g., by using `interp1` with the option `'cubic'` [1]). Choose the grid spacing and time step so that the numerical approximation is correct in the "eye norm", and submit your code under the file name `yourfamilyname_problem3b.m` .

(c) Now consider the advection-diffusion-reaction equation

$$\phi_t + u\phi_x = \mu\phi_{xx} + g(\phi) ,$$

where all parameters and functions (other than μ) are identical to the problem above. Write a program that numerically approximates the solutions to this problem and that is stable when choosing $\Delta t = \Delta x$. Demonstrate the performance of your method for two appropriate choices of μ: one choice where diffusion is small (but noticeable); and another choice where diffusion is large (but not too large). Submit your code under the file name `yourfamilyname_problem3c.m` . For full score it suffices to produce an overall first-order method. You receive **10 bonus points**, if the implemented method is at least second order and that fact is demonstrated via a numerical error convergence plot.

[1] As an optional side problem, try to understand what the difference between the options `'cubic'` and `'spline'` is; and argue which choice is preferable for the task at hand.