18.086 Pset #3

Due before class starts on April 8. Late homework will not be given any credit. Indicate on your report whether you have collaborated with others and whom you have collaborated with.

1. (**20 points**) *Scalar Conservation Laws.* Consider the initial value problem (IVP) of nonlinear conservation laws with proper boundary conditions:

$$u_t + f(u)_x = 0 \quad u(x,0) = u_0(x).$$

A first order conservative finite volumn scheme can be written as

$$u_j^{n+1} = u_j^n - \frac{\Delta t}{\Delta x}\left[\hat{f}\left(u_j^n, u_{j+1}^n\right) - \hat{f}\left(u_{j-1}^n, u_j^n\right)\right], \tag{0.1}$$

where $\hat{f}$ is the numerical flux. We regard u_j^n as an approximation to the cell average of $u(x,t)$ on the interval $I_j = [x_{j-\frac{1}{2}}, x_{j+\frac{1}{2}}]$. Up to second order accuracy, we can also regard u_j^n as an approximation to the point value at $x_j = \frac{x_{j-\frac{1}{2}}+x_{j+\frac{1}{2}}}{2}$. Examples of the numerical flux:

(1) *Godunov flux*: $\hat{f}(u^-, u^+) = f(u^*)$ where u^* is the value of the true solution at $x = 0$ of the Riemann problem, which refers to IVP for the same equation with piece-wise constant initial data: $u_0(x) = u^-$ if $x < 0$ and $u_0(x) = u^+$ if $x > 0$.

(2) *Lax-Friedrichs flux*: $\hat{f}(u^-, u^+) = \frac{1}{2}[f(u^-)+f(u^+)-\alpha(u^+-u^-)]$, where $\alpha = \max_u |f'(u)|$. The maximum is taken over u_j^n for all j.

The time step constraint of the two schemes is

$$\alpha\frac{\Delta t}{\Delta x} \le 1, \quad \alpha = \max_u |f'(u)|. \tag{0.2}$$

a (**5 points**) *Godunov flux.* Write down the explicit formula of the Godunov flux $\hat{f}(u^-, u^+)$ for Burgers' equation.

b (**5 points**) *Lax-Friedrichs flux.* Show that the Lax-Friedrichs flux $\hat{f}(u^-, u^+)$ with the time step constraint is monotone: $\hat{f}$ is non-decreasing w.r.t. the first input and non-increasing w.r.t. the second one. Denote the right hand side of (0.1) as a function G, i.e., $u_j^{n+1} = G(u_{j-1}^n, u_j^n, u_{j+1}^n)$. Show that G is non-decreasing w.r.t. all inputs. Thus the Lax-Friedrichs scheme satisfies the maximum principle: if $m \le u_j^n \le M$ for all j, then

$$m = G(m,m,m) \le u_j^{n+1} \le G(M,M,M) = M,$$

which is also true for the Godunov scheme. The exact solution has maximum principle: without boundary conditions (i.e., periodic or on a infinite domain), $m \le u(x,t) \le M$ for any $t > 0$, where m and M are min/max of the initial data. If there are nontrivial inflow boundary conditions, we need to add the effect of boundary conditions to the lower/upper bounds.

c (**10 points**)*Burgers' Equation.* Implement Godunov and Lax-Friedrichs schemes for Burgers' equation using the initial data $u_0(x) = \frac{1}{4} + \frac{1}{2}\sin(\pi x)$ on $x \in [-1, 1]$ with periodic boundary conditions. For convenience, we view u_j as midpoint values at x_j for each interval. Use the code on stellar for Problem 3 in Pset 2 to obtain the exact solution. Run the schemes till $T = 0.3$ with mesh points $N = 80, 160, 320, 640, 1280$. List the same error and order tables (two tables for two schemes) for the two schemes as in Problem 2 of Pset 2. Then run the schemes till $T = 2$ with $N = 160$ and plot two figures for the two schemes respectively: in each figure, plot the numercial solution as symbols and the exact solution as a solid curve. ***Remark:*** *generally speaking, the maximum wave speed* $\alpha = \max|f'(u)|$ *should be computed at every time step* t^n *and the max is taken over* u_j^n *for all* j*. But for this case, we have discrete maximum principle so we know* $-0.25 \le u_j^n \le 0.75$ *for any* n*. Thus we can use a constant time step by (0.2) with* $\alpha = \max_{u\in[-0.25,0.75]}|f'(u)|$*. However, this might be an overestimate of* α *for* $T = 2$ *(Why?).*

2. (**Bonus: 10 points**) *Nonlinear System of Conservation Laws: Compressible Euler Equations.* The 1D equations are given by

$$\mathbf{w}_t + \mathbf{f}(\mathbf{w})_x = \mathbf{0}, \quad t \ge 0,\ x \in \mathbb{R}, \tag{0.3}$$

$$\mathbf{w} = \begin{pmatrix} \rho \\ m \\ E \end{pmatrix}, \quad \mathbf{f}(\mathbf{w}) = \begin{pmatrix} m \\ \rho u^2 + p \\ (E+p)u \end{pmatrix}. \tag{0.4}$$

Here ρ is the density, u is the velocity, $m = \rho u$ is the momentum, E is total energy and p is pressure. We consider the perfect gas, for which the pressure is given by the *equation of state* $p = (\gamma - 1)(E - \frac{1}{2}\rho u^2)$ where $\gamma = 1.4$ is the adiabatic constant. The eigenvalues of the Jacobian $\mathbf{f}'(\mathbf{w})$ are u, $u \pm c$ where $c = \sqrt{\gamma \frac{p}{\rho}}$ is the sound speed. The Lax-Friedrichs scheme can be written similarly to (0.1) as:

$$\mathbf{w}_j^{n+1} = \mathbf{w}_j^n - \frac{\Delta t}{\Delta x}\left[\hat{\mathbf{f}}_{j+\frac{1}{2}} - \hat{\mathbf{f}}_{j-\frac{1}{2}}\right], \tag{0.5}$$

$$\hat{\mathbf{f}}_{j+\frac{1}{2}} = \frac{1}{2}[\mathbf{f}(\mathbf{w}_j^n) + \mathbf{f}(\mathbf{w}_{j+1}^n) - \alpha(\mathbf{w}_{j+1}^n - \mathbf{w}_j^n)], \tag{0.6}$$

where $\alpha = \max(|u| + c)$ is the maximum of wave speed and the max is taken over $\mathbf{w}_j^n$ for all j. $\hat{\mathbf{f}}_{j-\frac{1}{2}}$ is defined by shifting everything to the left by one. For systems of conservation laws, we no longer have the maximum principle. The velocity and sound speed could grow in time thus α varies every time step. Here is how to implement the scheme:

(1) Given $\mathbf{w}_j^n$, first compute the velocity by $u = m/\rho$, then compute the pressure by the equation of state and the sound speed.

(2) Compute $|u| + c$ for each j, then take the max over all j to get α (since it depends on $\mathbf{w}_j^n$ for all j, let's denote it as α^n). At each time level t^n, we need a new time step, which is determined by $\alpha^n \frac{\Delta t}{\Delta x} = 1$.

(3) Compute the numerical flux (0.6) then compute $\mathbf{w}_j^{n+1}$ by (0.5).

a (**Bonus: 5 points**) Implement the Lax-Friedrichs scheme defined above. Use the following exact solution to check the accuracy. The initial condition is

$$\rho_0(x) = 1 + 0.5\sin(x),\ u_0(x) = 1,\ p_0(x) = 1.$$

The domain is $x \in [0, 2\pi]$ and the boundary condition is periodic. The exact solution is

$$\rho(x,t) = 1 + 0.5\sin(x-t),\ u(x,t) = 1,\ p(x,t) = 1.$$

Check the error and order for the density only at $T = 0.3$ and list them as a table for mesh points $N = 8, 16, 32, 64, 128, 256$.

b (**Bonus: 5 points**) *Riemann Problem: Sod's Problem.* The initial data is

$$\mathbf{w}(x,t) = \begin{cases} \mathbf{w}_l, & x < 0 \\ \mathbf{w}_r, & x > 0 \end{cases}$$

with $(\rho_l, u_l, p_l) = (1, 0, 1)$ and $(\rho_r, u_r, p_r) = (0.125, 0, 0.1)$ on $x \in [-5, 5]$. Use constant extensions as boundary conditions. Plot your numerical density at $T = 2$ with $N = 1000$ points and the exact one in one figure. Download the exact solution from http://www-math.mit.edu/classes/18.086/2014/files/Soddensity.zip

3. (**30-40 points**) *2D incompressible Navier-Stokes.* Consider the vorticity stream-function formulation:

$$\omega_t + u\omega_x + v\omega_x = \frac{1}{Re}\Delta\omega, \tag{0.7}$$

$$\Delta\psi = \omega, \qquad \langle u, v\rangle = \langle -\psi_y, \psi_x\rangle, \tag{0.8}$$

$$\omega(x, y, 0) = \omega_0(x, y)(\text{initial condition}), \qquad \langle u, v\rangle \cdot \mathbf{n} = \text{given on } \partial\Omega(\text{boundary condition}).$$

Here ψ is the stream function and ω is the vorticity, which is the curl of the velocity field $\mathbf{u} = \langle u, v\rangle$. For simplicity, we only consider periodic cases to bypass the boundary condition.

a *1D Poisson Equation with Periodic B.C.* Consider $-u_{xx} = f$ on $x \in [0, 2\pi]$ with periodic boundary conditions. First, if we assume u and u_x are periodic, then $\int_0^{2\pi} f(x)dx = 0$. Thus f must have average 0 to be consistent with the periodicity of u_x. Second, if we use the central difference for the periodic case, the eigenvectors of the corresponding negative discrete Laplacian matrix are precisely those in Discrete Fourier Transform with eigenvalues $2 - 2\cos(k\Delta x)$ for $k = 0, 1, \cdots, N-1$. Third, let $\hat{u}$ and $\hat{f}$ denote the discrete Fourier Transform, then the linear system of the central difference is equivalent to $(2 - 2\cos(k\Delta x))\hat{u}(k) = \hat{f}(k)$. For $k = 0$, if $\hat{f}(0) \neq 0$, then the system does not have a solution. But fortunately $\hat{f}(0)$ should be very close to zero numerically if $\int_0^{2\pi} f(x)dx = 0$ (why?). Last, the solution to this problem is not unique. Any solution plus a constant is still a solution. This is due to the periodic assumption which is as good as no boundary conditions. For the 2D Navier-Stokes vorticity equations, the non-uniqueness does not cause any confusion since we only need the partial derivatives of ψ to update ω. We simply set $\hat{u}(0)$ to be 0. For instance, we can solve the linear system by FFT in MATLAB as the following:

```
% 1D Poisson Eqn -u_xx=f with periodic b.c.
n=320; dx=2*pi/n; x=linspace(0,2*pi,n+1)';x=x(2:end); % mesh
f=-2*cos(2*x); fhat=fft(f); % the integral of f must be zero.
% Eigenvalues are 2-2*cos(k*dx) for k=0,1,...,n-1
eigenvalue=(2-2*cos(dx*[0:n-1]'))/dx^2;
% By setting the first eig to be 1, we set  uhat(1)=fhat(1).
eigenvalue(1)=1; uhat=fhat./eigenvalue;
u=real(ifft(uhat)); % u is the final numerical solution
true=sin(x).^2-1/2; error=max(abs(u-true))
```

b (**10 points**) *2D Fast Poisson Solver.* Extend the above code to 2D case for solving $u_{xx} + u_{yy} = f$ on $[0, 2\pi] \times [0, 2\pi]$ with periodic assumptions. Test your code with the following solution: $f = 2\cos(2x) + 2\cos(2y)$ and $u = \sin^2 x + \sin^2 y - 1$. Remember to set $\hat{u}(0, 0) = 0$ to get the correct solution. Use uniform $N \times N$ meshes. List a table with four columns: mesh point number, error, order and CPU time for $N = 320, 640, 1280, 2560, 5120$.

c (**10 points**) *Linear Convection Diffusion.* To solve the nonlinear convection diffusion, test the discretization on the linear one first. Consider the following equation with periodic b.c. on $[0, 2\pi]$:

$$u_t + u_x + u_y = d(u_{xx} + u_{yy}), \quad d > 0.$$

To have second order accuracy, the most straightforward choice is the central difference (which is a linear discretization thus easy to vectorize in MATLAB) for first and second order derivatives. Let h denote mesh size of a uniform mesh and Δ_h denote the discrete Laplacian:

$$\frac{d}{dt}u_{i,j} = -\frac{u_{i+1,j} - u_{i-1,j}}{2h} - \frac{u_{i,j+1} - u_{i,j-1}}{2h} + d\Delta_h u_{i,j}. \tag{0.9}$$

We may use implicit methods for the diffusion but efficiently finding the inverse of the resulting huge matrix in 2D is more complicated than 1D case. On the other hand, if we use explicit method for the Laplacian part, the time step constaint will be $\Delta t \sim \frac{h^2}{d}$ which is acceptable if d is very small (for instance $d \sim h$) as in high Reynolds number flow. However, using forward Euler or RK2 for (0.9) with $d = 0$ is never stable because their stability regions do not contain any imaginary axis and we get the factor $\mathbb{i}\frac{\sin(k\Delta x)}{\Delta x}$ after plugging ansatz $u_j = e^{\mathbb{i}kx_j}$ into $\frac{u_{j+1}-u_{j-1}}{2\Delta x}$. The stability regions of RK3 and RK4 contain part of the imaginary axis thus we could have a reasonable stable time step for small d.

Suppose $U(i, j)$ denotes the point value at (x_i, y_j), then the right hand side of (0.9) can be vectorized as:

```
% discrete Laplacian
 I = speye(n,n);
 E = sparse(2:n,1:n-1,1,n,n);
 D = E+E'-2*I;
 D(n,1)=1; D(1,n)=1;% for periodicity
```

```
 Laplacian = (kron(D,I)+kron(I,D))/dx^2;
% discrete Gradient
 D = -E+E';
 D(n,1)=1; D(1,n)=-1;% for periodicity
 Gradient = (kron(D,I)+kron(I,D))/dx/2;
 K=-Gradient+d*Laplacian;
 V=reshape(U,[],1);
 RHS=reshape(K*V,size(U));
```

Use the following Strong Stability Preserving (SSP) RK3 (also call TVD RK3) to solve (0.9) with the time step $\Delta t = \min\{0.3\frac{h^2}{d}, 0.3h\}$ ($h = \Delta x = \Delta y$):

```
    U1=U+dt*RHS(U);
    U2=0.75*U+0.25*(U1+dt*RHS(U1));
    U=U/3+2/3*(U2+dt*RHS(U2));
```

Test your code using the exact solution $u(x,t) = exp(-2dt)\sin(x+y-2t)$ with $d = 0.01$ on $[0, 2\pi] \times [0, 2\pi]$. Run it till $T = 0.2$ with $N = 16, 32, 64, 128, 256, 512$. Show the table of error and order.

d (**10 points**) *2D Incompressible Flow.* Plugging the second equation of (0.8) into (0.7), we get

$$\omega_t - \psi_y \omega_x + \psi_x \omega_y = \frac{1}{Re}\Delta\omega.$$

Let D_x and D_y denote the central difference for the first order partial derivatives, we get

$$\omega_t = D_y\psi D_x\omega - D_x\psi D_y\omega + \frac{1}{Re}\Delta_h\omega.$$

Use the RK3 above for the time derivative. At each time step t^n, first compute the maximum of velocity by $U^n = \max\{|u^n|, |v^n|\}$, then time step can be taken as $\Delta t = \min\{0.3 * Re * h^2, 0.3\frac{h}{U^n}\}$ ($h = \Delta x = \Delta y$). Test the accuracy with the exact solution: $\omega(x, y, t) = -2\exp(-2t/Re)\sin x \sin y$ on domain $[0, 2\pi] \times [0, 2\pi]$ with $Re = 100$. Run your code till $T = 0.2$ with $N = 16, 32, 64, 128, 256$. List the error and order as a table.

e (**Bonus: 10 points**) *2D Incompressible Flow: Double Shear Layer.* Take $Re = 1000$. The initial condition is

$$\omega(x, y, 0) = \begin{cases} \delta\cos(x) - \frac{1}{\rho}\text{sech}^2((y - \pi/2)/\rho) & y \le \pi \\ \delta\cos(x) + \frac{1}{\rho}\text{sech}^2((3\pi/2 - y)/\rho) & y > \pi \end{cases}$$

on domain $[0, 2\pi] \times [0, 2\pi]$, where we take $\rho = \pi/15$ and $\delta = 0.05$. Show your vorticity at $T = 6$ and $T = 8$ with $N = 256$ using 30 contour lines by the command

```
% if omega(i,j) denotes the vorticity at (x_i,y_j)
 contour(omega,30); colorbar
```

Make sure your x-axis and y-axis are correctly shown.