We can check that $DD^T = K$ with K being the tridiagonal (-1, 2, -1) matrix, which is the negative discrete Laplacian matrix in Chapter 2. This means that the Laplacian operator is related to enforcing the incompressible condition.

4.1.3 *Project: incompressible flows with periodic boundary*

Consider the vorticity stream-function formulation of the *2D incompressible Navier-Stokes* given by (4.3) on a square domain with initial condition $\omega(x, y, 0) = \omega_0(x, y)$ and periodic boundary conditions.

Write a MATLAB code for solving this equation by following the instructions below step by step.

Step I: 2D Poisson Solver. We need to solve a Poisson equation $\Delta\psi = \omega$. Implement the eigenvector method in Chapter 2 for solving $u_{xx}+u_{yy} = f$ on $[0, 2\pi] \times [0, 2\pi]$ with periodic boundary conditions. Test your code with the following solution: $f = 2\cos(2x) + 2\cos(2y)$ and $u = \sin^2 x + \sin^2 y - 1$. Recall that the matrix is singular thus we usually redefine the division by zero eigenvalue as multiplying zero, then we can compare it with an exact solution which sums to zero. Use *fft2* and *ifft2* functions for the eigenvector multiplication. Use uniform $N \times N$ meshes. Show loglog plot of the errors in max norm and compare it with the second order slope line for $N = 20, 40, 80, 160, 320$. Show your CPU time (use tic, toc functions in MATLAB to track CPU time) for $N = 1280, 2560, 5120$.

Step II: Linear Convection Diffusion. To solve the nonlinear convection diffusion like Navier-Stokes equation, we should first test our numerical scheme on a linear equation. Consider the following equation with periodic b.c. on $[0, 2\pi] \times [0, 2\pi]$:

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

We can use centered difference for all spatial derivatives to achieve second order accuracy in space. Let $h = \Delta x = \Delta y$ denote mesh size of a uniform mesh and Δ_h denote the discrete Laplacian:

$$\frac{d}{dt}u_{i,j} = -\frac{u_{i+1,j} - u_{i-1,j}}{2h} - \frac{u_{i,j+1} - u_{i,j-1}}{2h} + d\Delta_h u_{i,j}. \tag{4.4}$$

If we use explicit methods like forward Euler or second order Runge Kutta (RK) method for (4.4) with $d = 0$, it is unstable for any time step because their stability regions do not contain any imaginary axis. Instead, higher

order accurate RK methods will work well with a reasonable stable time step for small d. If we write the ODE system (4.4) as $\frac{d}{dt}U = RHS$, use the following Strong Stability Preserving (SSP) RK3 (third order RK) to solve $\frac{d}{dt}U = RHS$ 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 $N = 16, 32, 64, 128, 256, 512$. If the error has the order $e(h) = Ch^k$ on a mesh with size h, then on a refined mesh we have $e(h/2) = C(h/2)^k$. Therefore, we have

$$\log e(h/2) = \log C + k\log h - k\log 2, \quad \log e(h) = \log C + k\log h$$

thus we can compute the convergence order using the errors $e(h), e(h/2)$:

$$k = \frac{\log e(h) - \log e(h/2)}{\log 2} = \frac{1}{\log 2}\log\frac{e(h)}{e(h/2)}. \tag{4.5}$$

Show a table of errors and orders computed by refining the mesh: we have six errors on six meshes so we can compute 5 orders by (4.5).

Step III: Accuracy Test for 2D Incompressible Flow. Plugging the second equation of (4.3b) into (4.3a), we get

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

Let D_x and D_y denote the central difference operator for the first order partial derivatives, and let Δ_h be the discrete Laplacian using second order finite difference, we get

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

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 $\Delta t = \min\{0.3\,\mathrm{Re}\,h^2, 0.3\frac{1}{U^n}h\}$ ($h = \Delta x = \Delta y$). For each time stage, you need to compute/update the velocity by solving the Poisson equation ψ. Test the accuracy with the exact solution: $\omega(x,y,t) = -2\exp\{(-2t/\,\mathrm{Re})\}\sin x\sin y$ on domain $[0, 2\pi] \times [0, 2\pi]$ with $\mathrm{Re} = 100$. Notice that the 2D Poisson equation needs to be solved for each of the

three time stages in one RK step. Run your code till $T = 0.2$ with $N = 16, 32, 64, 128, 256$. List the error and order as a table.

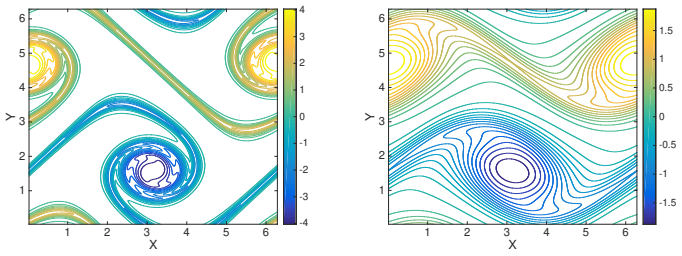
Step IV: Double Shear Layer Benchmark Test. Once you have finished the three steps above, you can change your initial condition to further test your code on some benchmark tests. Take Re $= 1000$. The initial condition is

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

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

For example, the initial value can be visualized using 30 contour lines on a 256×256 mesh as follows. See also Figure 4.1 for the solution at $T = 8$.

```
n=256; L  = 2*pi;
x  = linspace(0,L,n+1)'; x = x(2:end); y=x;
yy=y*ones(1,n); xx=ones(n,1)*x';
rho=pi/15; Delta=0.05;
omega1=Delta*cos(xx)-1/rho*sech((yy-pi/2)/rho).^2;
omega2=Delta*cos(xx)+1/rho*sech((3*pi/2-yy)/rho).^2;
indicator1=[ones(n/2,n); zeros(n/2,n)];
indicator2=[zeros(n/2,n); ones(n/2,n)];
omega=omega1.*indicator1+omega2.*indicator2;
contour(x,y,omega,30);colorbar; xlabel('X');ylabel('Y');
```



(a) Re $= 1000$. $T = 8$. 256×256 grid. (b) Re $= 70$. $T = 8$. 512×512 grid.

Fig. 4.1 Incompressible Navier Stokes equations: Double Shear Layer.

Step V: Implicit diffusion treatment. For small Re, the time step $\Delta t = 0.3\,\mathrm{Re}\,h^2$ is too small to use. Instead, we can consider using the implicit-explicit (IMEX) method, i.e., consider the following scheme which is first order accurate in time:

$$\omega^{n+1} = \omega^n - \Delta t u^n D_x \omega^n - \Delta t v^n D_y \omega^n + \frac{\Delta t}{\mathrm{Re}} \Delta_h \omega^{n+1},$$
$$u^n = -D_y \psi^n, \quad v^n = D_x \psi^n, \quad \Delta_h \psi^n = \omega^n.$$

Such an IMEX method allows a larger time step. At each time step, define $\|\mathbf{u}^n\|_\infty = \max\{\max_{i,j} |u^n_{i,j}|, \max_{i,j} |v^n_{i,j}|\}$. Then the linear stability analysis, which is not covered in this book, suggests that the IMEX scheme is linearly stable if

$$\Delta t \le \frac{1}{\|\mathbf{u}^n\|^2_\infty} \frac{1}{\mathrm{Re}}. \tag{4.6}$$

Implement this scheme for the Double Shear Layer problem using a time step as $\Delta t = \Delta x$ for $\mathrm{Re} = 70$. Use the eigenvector method to invert the matrix and use FFT for the eigenvectors. Show your vorticity at $T = 6$ and $T = 8$ with $N = 512$ using 30 contour lines. You can also try a larger Δt or larger Re so that (4.6) is violated on a coarse mesh, and see what happens.

Bonus: high order accuracy by Q^2 finite element method. Follow [Shen and Zhang (2021)] to use the fourth order finite difference therein, which is the Q^2 finite element method with Simpson quadrature (e.g., read Section 3.12). Implement it for the double shear layer problem and compare your results to those in [Shen and Zhang (2021)]. Notice that Section 3.12 is for Dirichlet b.c., and periodic boundary conditions are needed here.

4.1.4 *Project: Lid Driven Cavity flow*

The Lid Driven Cavity flow is a classical test problem for numerical methods solving incompressible Navier-Stokes equations. The domain Ω is a square $[0,1] \times [0,1]$ and the boundary conditions of the velocity is specified as in Figure 4.2. In other words, the velocity along the boundary is zero, except that u is equal to some given velocity at the top boundary.

Consider the vorticity stream-function formulation of the *2D incompressible Navier-Stokes* given by (4.3) on a square domain $[0,1]^2$ with the initial stream function $\psi_0(x,y) = 16(y^2 - y^3)x^2(1-x)^2$ and the initial vorticity

$$\omega_0(x,y) = \frac{\partial^2}{\partial x^2}\psi_0 + \frac{\partial^2}{\partial y^2}\psi_0.$$