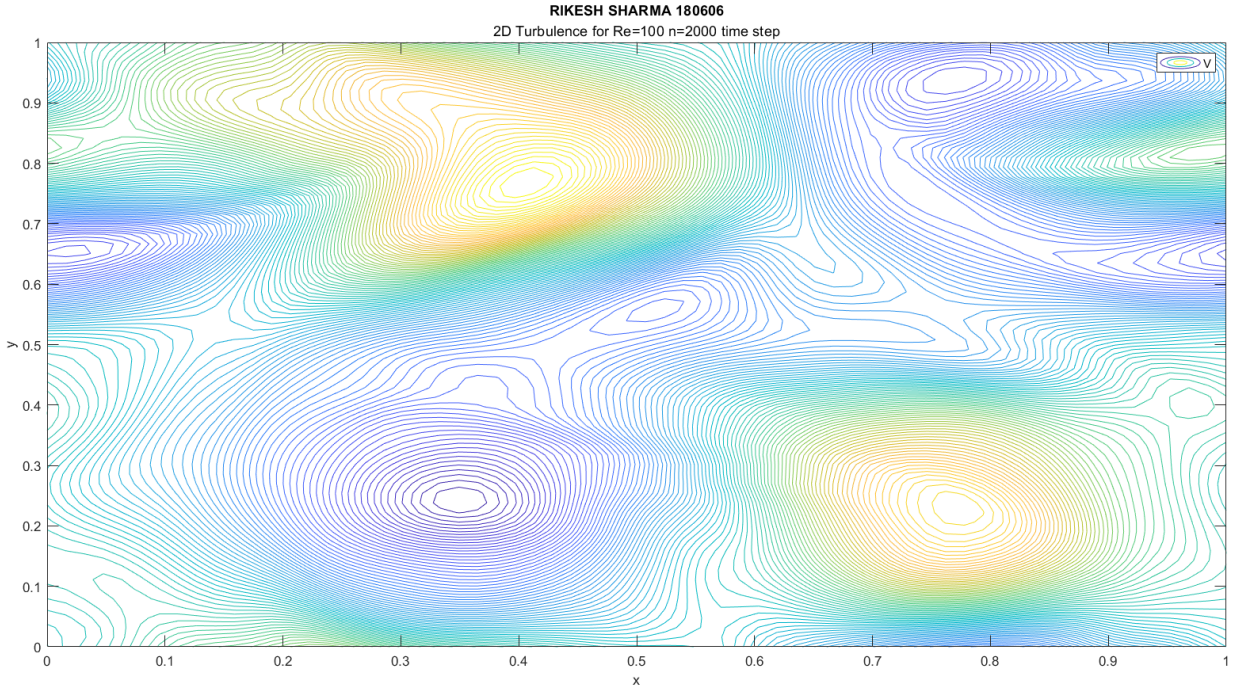
ME630A



Assignment 4

(20 November 2021)

Rikesh Sharma

180606

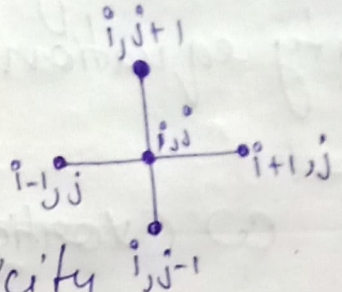
ME630
Assignment-4

Given

$L_x = L_y = 1$ for $N_x \times N_y = 64 \times 64$

with periodic Boundary Conditions

$Re = 100$ and $Re = 1000$



To solve Navier-Stokes equation using stream-Vorticity function approach for 2D-Turbulence problem. incompressible.

We know the governing Equations in 2D-incompressible flow are — (in non-dimensionalized form)

Continuity: $\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0$ — (1)

x-momentum: $\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{\partial P}{\partial x} + \frac{1}{Re}\left\{\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right\}$ — (2)

y-momentum: $\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{\partial P}{\partial y} + \frac{1}{Re}\left\{\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right\}$ — (3)

We introduce $\psi \equiv$ Stream function

Such that $u = \frac{\partial \psi}{\partial y}$ — (4) $v = -\frac{\partial \psi}{\partial x}$ — (5)

Using eqn ④ and eqn ⑤ in eqn ①

$$\frac{\partial}{\partial x}\left(\frac{\partial \psi}{\partial y}\right) + \frac{\partial}{\partial y}\left(-\frac{\partial \psi}{\partial x}\right) = 0$$

Stream function ψ naturally satisfies Continuity equation.

now ω (vorticity) $\equiv \frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}$ — ⑥

now taking $\frac{\partial}{\partial x}$ [Equation 3] $-$ $\frac{\partial}{\partial y}$ [Equation 2]

and simplifying we get —

— Vorticity transport Equation as

$$\boxed{\frac{\partial \omega}{\partial t} + u\frac{\partial \omega}{\partial x} + v\frac{\partial \omega}{\partial y} = \frac{1}{Re}\left\{\frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2}\right\}} \quad — ⑦$$

Also we know $\omega = \frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}$ and $u = \frac{\partial \psi}{\partial y}$

and $v = -\frac{\partial \psi}{\partial x}$

Therefore

Replacing u and v in eqn of ω.

we get $\boxed{\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -\omega}$ —— ⑧

Now we have to solve eq (7) and eq (8) simultaneously.

From eq^n (7) we can write

$$\frac{\partial \omega}{\partial t} = -u\frac{\partial \omega}{\partial x} - v\frac{\partial \omega}{\partial y} + \frac{1}{Re}\left\{\frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2}\right\}$$

Which is of the form.

$\frac{\partial \phi}{\partial t} = f(t, \phi)$ and can be solved using RKW3.

From eq^n (8) we can write

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -\omega$$

which has the form.

of $\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = S_\psi$ {Pressure-poisson eq^n

and we can use Gauss-Siedel to solve it.

1) Discretization

(A) we have from eq (7)

$$\frac{\partial \omega}{\partial t} = -u\frac{\partial \omega}{\partial x} - v\frac{\partial \omega}{\partial y} + \frac{1}{Re}\left\{\frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2}\right\}$$

Assuming uniform grids i) $\Delta x_k = \Delta x_m = \Delta x$
ii) $\Delta y_k = \Delta y_m = \Delta y$
iii) and $\Delta x = \Delta y$

$$\frac{\partial \omega}{\partial t} = -u_{ij}^n\left(\frac{\omega_{i+1,j}^n - \omega_{i-1,j}^n}{2\Delta x}\right) - v_{ij}^n\left(\frac{\omega_{i,j+1}^n - \omega_{i,j-1}^n}{2\Delta y}\right)$$
$$+\frac{1}{Re}\left\{\frac{\omega_{i+1,j}^n - 2\omega_{i,j}^n + \omega_{i-1,j}^n}{\Delta x^2} + \frac{\omega_{i,j+1}^n - 2\omega_{i,j}^n + \omega_{i,j-1}^n}{\Delta y^2}\right\}$$

~~$\frac{\partial \omega}{\partial t} = f(t_n, \omega_n)$~~
~~Since $f(t_n, \omega_n) \neq -u_{ij}$~~

writing it as.

$$\frac{\partial \omega}{\partial t} = f(t_n, \omega_n)$$

we have

$$f(t_n, \omega_n) = -u_{ij}^n\left(\frac{\omega_{i+1,j}^n - \omega_{i-1,j}^n}{2\Delta x}\right) - v_{ij}^n\left(\frac{\omega_{i,j+1}^n - \omega_{i,j-1}^n}{2\Delta y}\right)$$
$$+\frac{1}{Re}\left\{\frac{\omega_{i+1,j}^n - 2\omega_{i,j}^n + \omega_{i-1,j}^n}{\Delta x^2} + \frac{\omega_{i,j+1}^n - 2\omega_{i,j}^n + \omega_{i,j+1}^n}{\Delta y^2}\right\}$$

∴ Using RKW3 (psued code)

1st RK $\omega^* = \omega^n$

$K_1 = f(t_n, \omega^*)$

2nd RK $\omega^* = \omega^* + \frac{\Delta t}{3} K_1$

$K_1 = -\frac{5}{9} K_1 + f(t_n + \frac{\Delta t}{3}, \phi^*)$

3rd RK $\omega^* = \omega^* + \frac{15}{16} \Delta t K_1$

$K_1 = -\frac{153}{128} K_1 + f(t_n + \frac{3\Delta t}{4}, \omega^*)$

$\omega^{n+1} = \omega^* + \frac{8}{15} \Delta t K_1$

and Repeat for all time steps required.

(B) from eqn (8)

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -\omega$$

Assume uniform grids

$\Delta x_K = \Delta x_m = \Delta x$

$\Delta y_K = \Delta y_m = \Delta y$

$\Delta x = \Delta y$

Discretizing

$$\frac{\psi_{i+1,j} - 2\psi_{i,j} + \psi_{i-1,j}}{\Delta x^2} + \frac{\psi_{i,j+1} - 2\psi_{i,j} + \psi_{i,j-1}}{\Delta y^2} = -\omega_{i,j}^{n+1}$$

Rearranging and simplifying

$$\psi_{i,j} = \frac{\omega_{ij} + \left\{ \frac{\psi_{i,j-1}}{\Delta y^2} + \frac{1}{\Delta x^2}\psi_{i-1,j} + \frac{1}{\Delta x^2}\psi_{i+1,j} + \frac{\psi_{i,j+1}}{\Delta y^2} \right\}}{2\left\{ \frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} \right\}}$$

Since grid is uniform and we will be using Gauss-Siedel method =)

Above eqn simplifies to, "for K^{th} iteration"

$$\psi_{i,j}^{K+1} = \frac{1}{4}\left\{ \psi_{i+1,j}^{K} + \psi_{i-1,j}^{K+1} + \psi_{i,j+1}^{K} + \psi_{(i,j-1)}^{K+1} + \omega^{n+1}(i,j)\,\Delta x^2 \right\}$$

Where

$\psi_{i,j}^{K+1}$ is the value of ψ at $(i,j)^{th}$ location and after K^{th} iteration in Gauss-Siedel

$\omega_{i,j}^{n+1}$ is the value of ω at $(i,j)^{th}$ location and at n^{th} time step.

Hence ~~to~~ Now we can solve eq^n (7) and eq^n (8) simultaneously.

Psuedocode for complete solution

1. Define the domain $Lx = 1$ $Ly = 1$
2. Define no of grids points $Nx = 64$ $Ny = 64$
   $nxp2 = Nx + 2$ $nyp^2 = Ny + 2$.
3. Initialize time domain.
4. Initialize ψ using random numbers $-1 < rm < 1$ over entire domain.
5. Calculate u and V using

$$u_{ij}^n = (\psi_{i=,j+1}^n \; \psi_{i,j-1}^n)/2\Delta x \qquad V_{ij}^n = \frac{-(\psi_{i+1,j}^n - \psi_{i-1,j}^n)}{2\Delta y}$$

6. Initialize ~~Omega~~ ω using the values obtained from u and v at $t = 0.0$

using $\psi_{ij}^{n} = \left(\frac{v_{i+1,j} - v_{i-1,j}}{2\Delta x} - \frac{u_{i,j+1} - u_{i,j-1}}{2\Delta y} \right)$

7. Define $Re = 100$

8. Now we know ψ_0, u_0, v_0 and ω_0

9. Start a for loop for $n = 1$ to N_{max}

9.1 use RKW3 as discussed in discretization of $\left(\frac{d\omega}{dt} = f(t_n, \omega_n)\right)$ to find ω^{n+1}

9.2 use Gauss Siedel as discussed to Solve for ψ^{n+1} using ω^{n+1}

9.3 use ψ^{n+1} to get u^{n+1} and v^{n+1}

9.4 Update boundary conditions

$u(N_x+2,:) = u(2,:)$

$u(1,:) = u(N_x+1,:)$

$u(:,N_x+2) = u(:,2)$

$u(:,1) = u(:,N_x+1)$

$v(N_x+2,:) = v(2,:)$

$v(1,:) = v(N_x+1,:)$

$v(:,N_x+2) = v(:,2)$

$v(:,1) = v(:,N_x+1)$

$\omega(N_x+2,:) = \omega(2,:)$

$\omega(1,:) = \omega(N_x+1,:)$

$\omega(:,N_x+2) = \omega(:,2)$

$\omega(:,1) = \omega(:,N_x+1)$

$\psi(N_x+2,:) = \psi(2,:)$

$\psi(1,:) = \psi(N_x+1,:)$

$\psi(:,N_x+2) = \psi(:,2)$

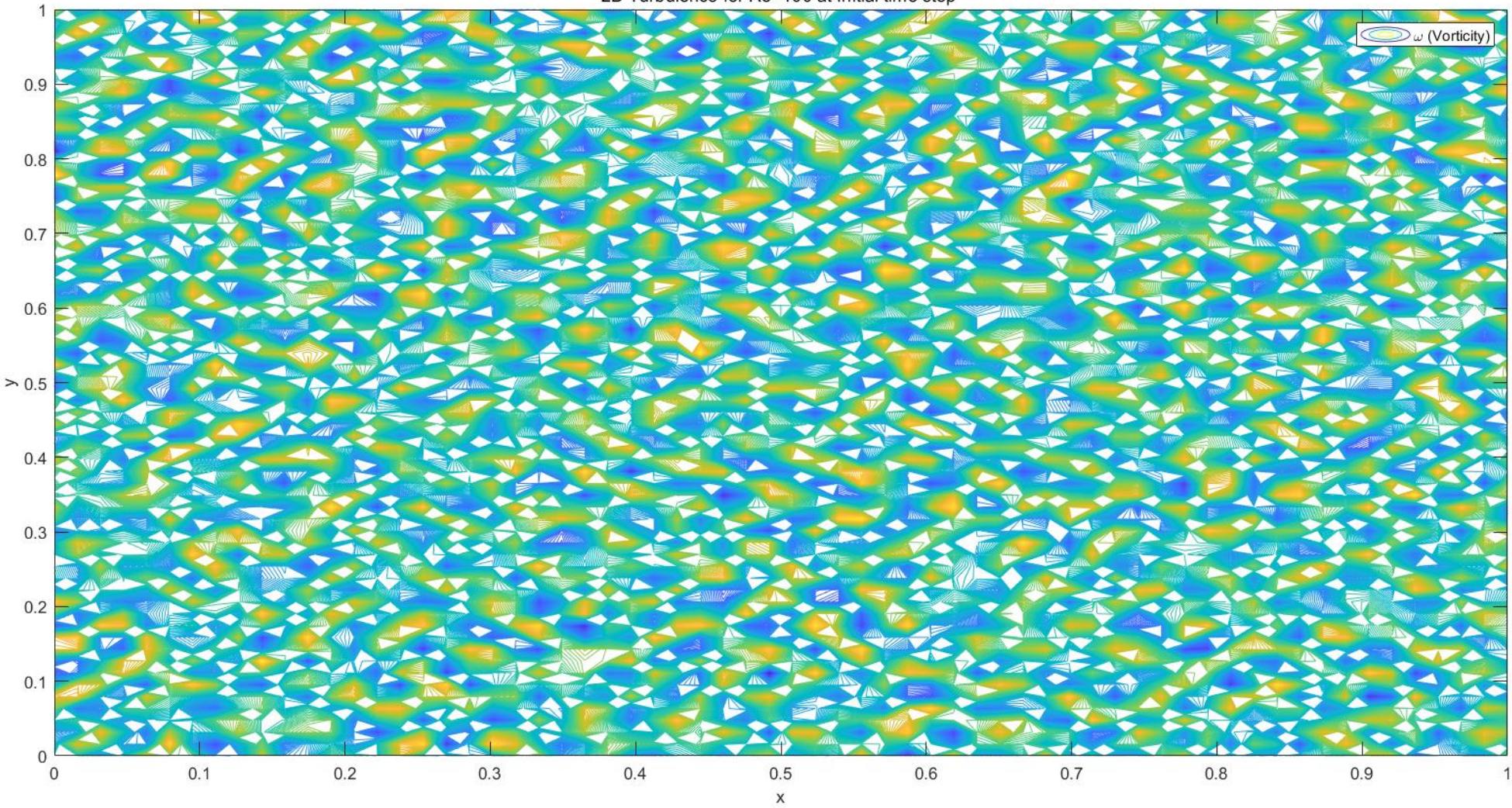
$\psi(:,1) = \psi(:,N_x+1)$

10. Repeat and Save images to form a movie of the evolution of 2D turbulence / Vorticity

11. Write the movie to a file.

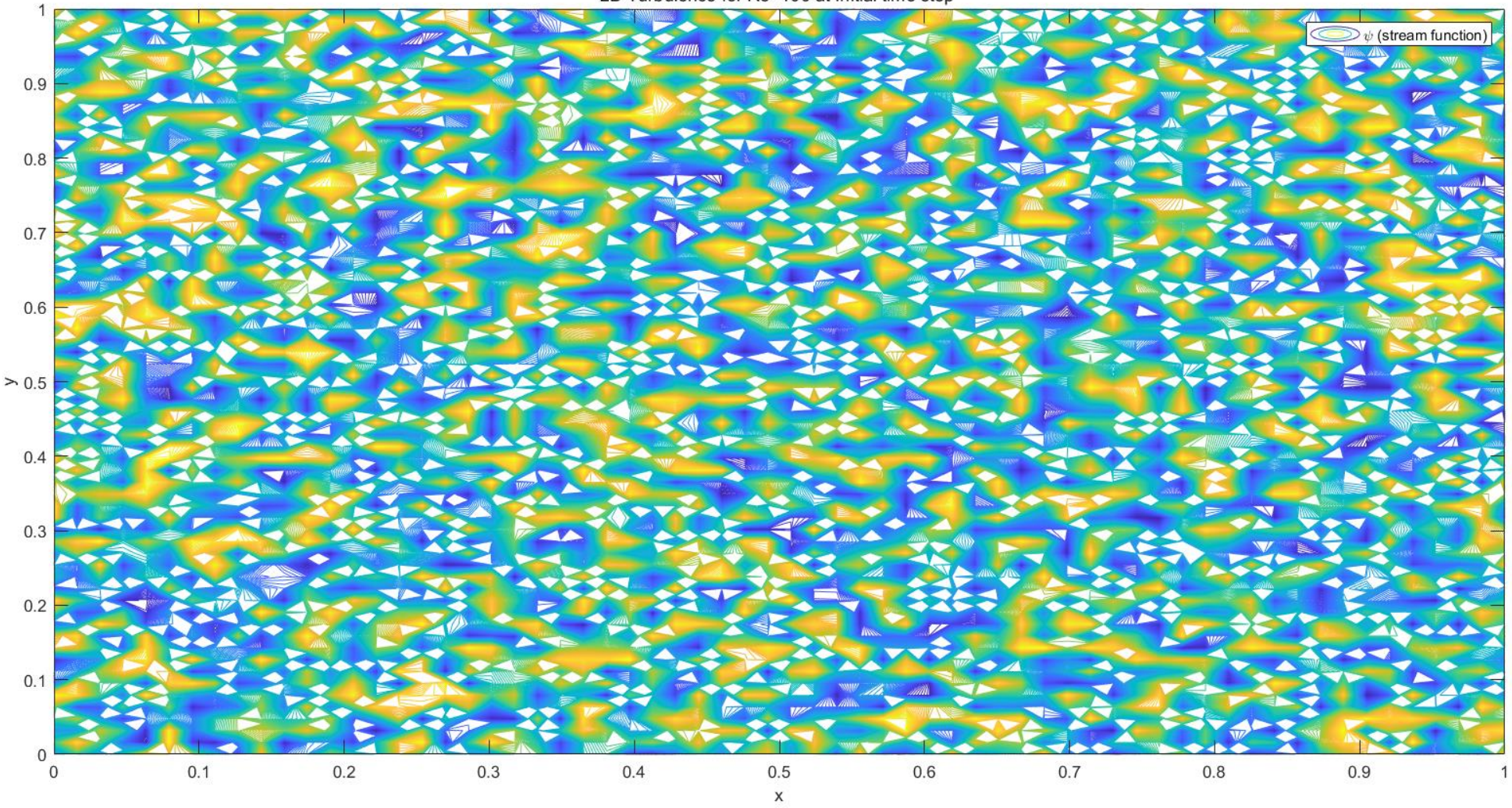
RIKESH SHARMA 180606

2D Turbulence for Re=100 at Initial time step



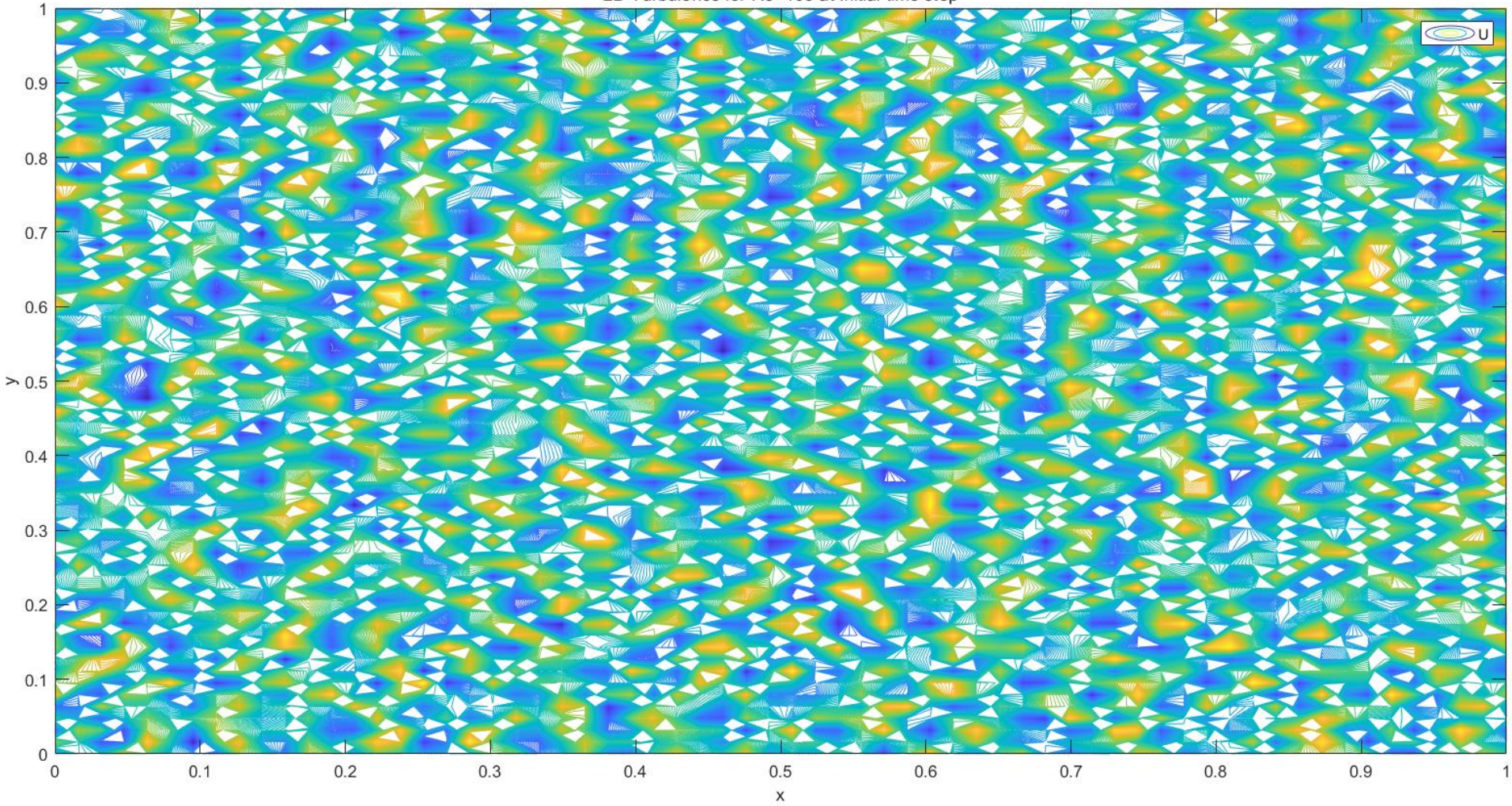
RIKESH SHARMA 180606

2D Turbulence for Re=100 at Initial time step



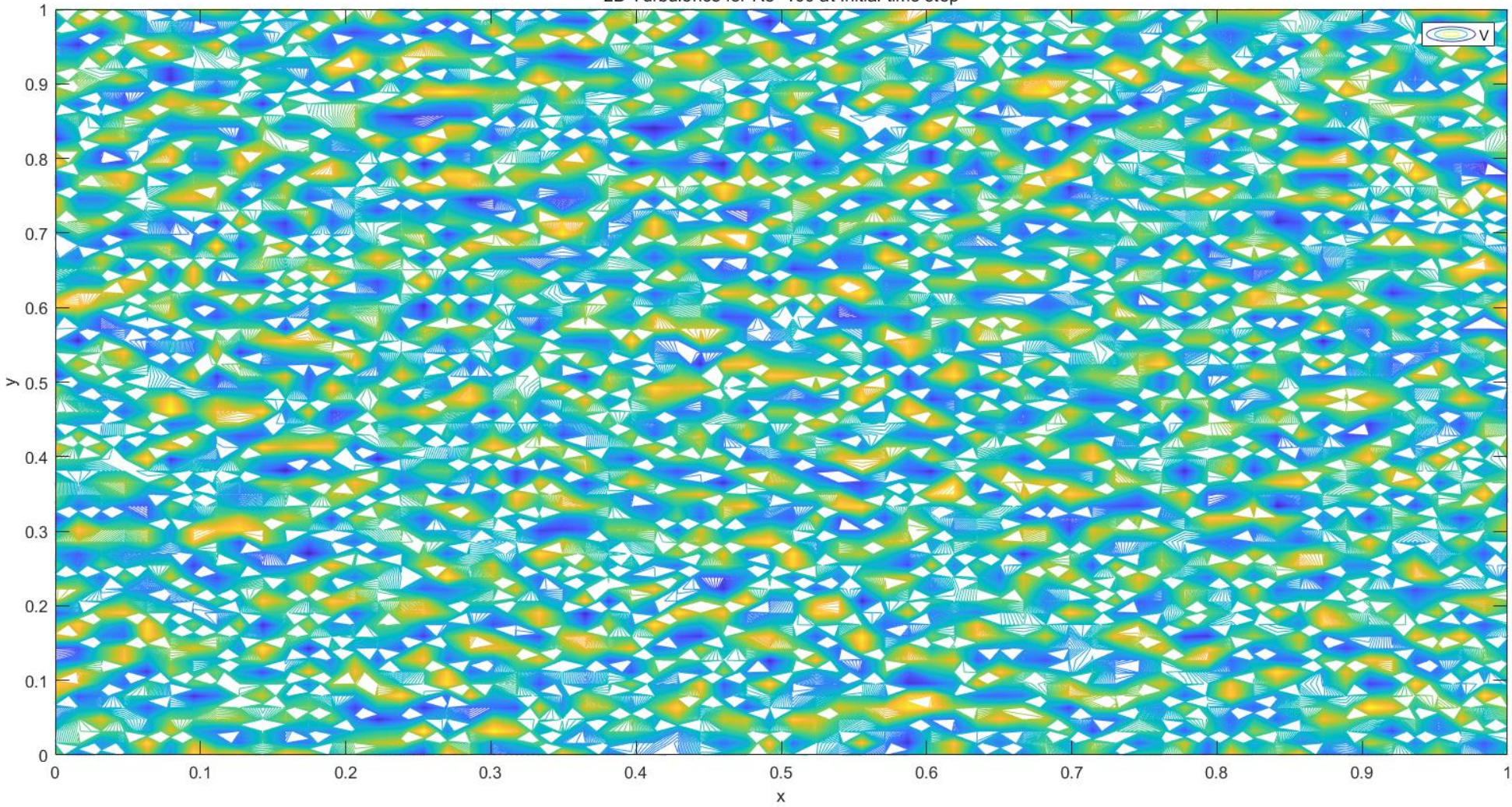
RIKESH SHARMA 180606

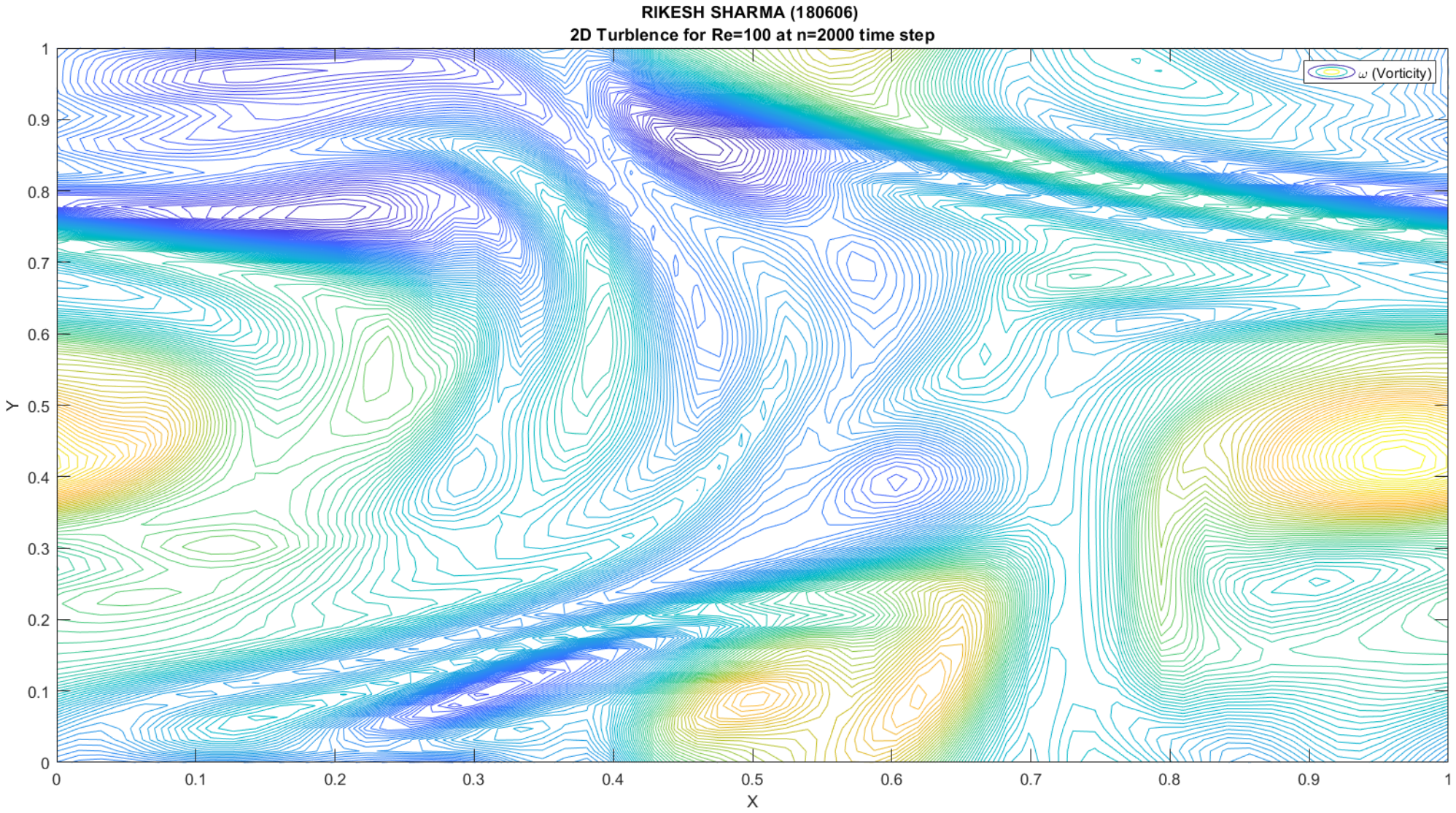
2D Turbulence for Re=100 at Initial time step



RIKESH SHARMA 180606

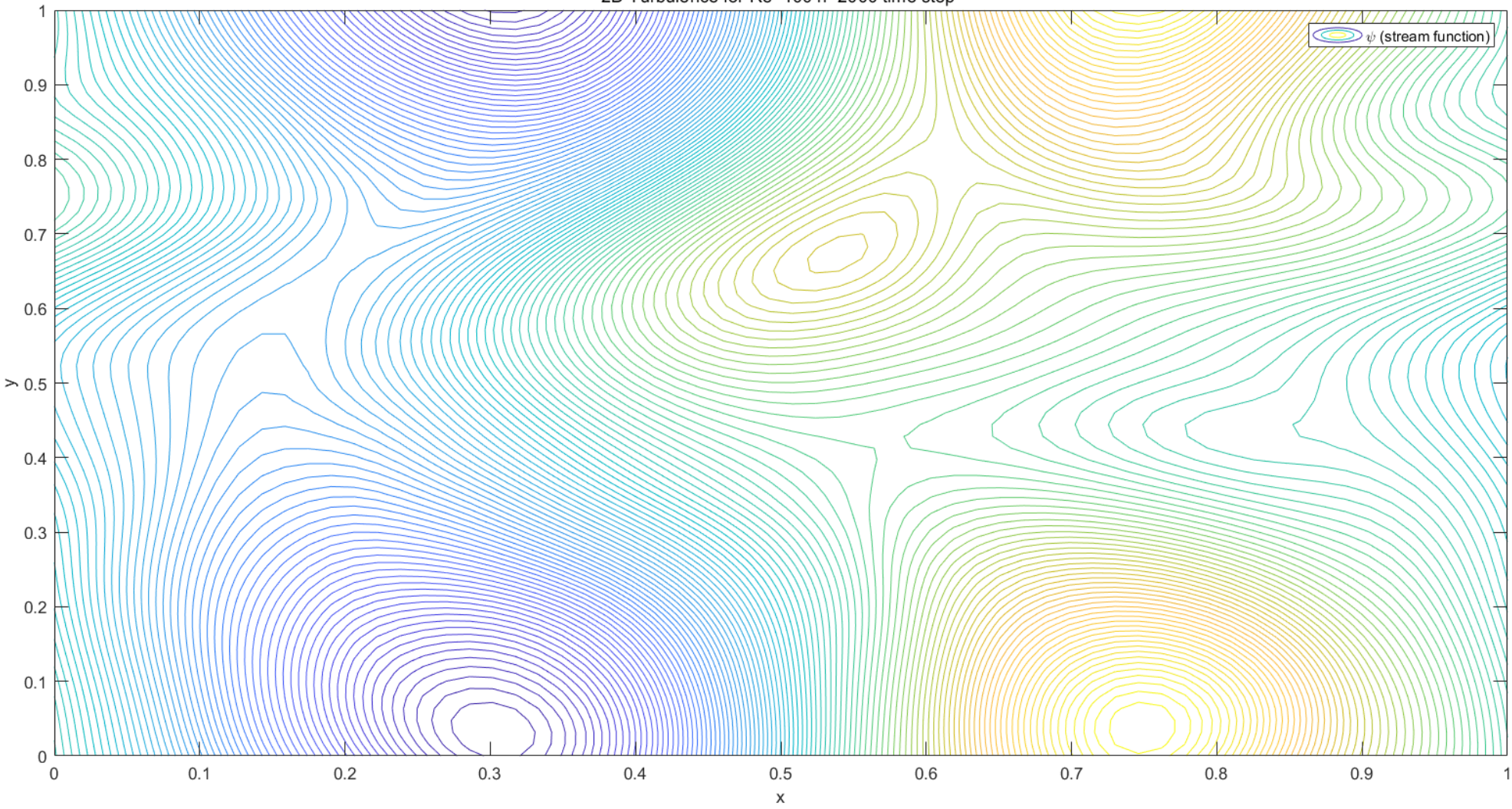
2D Turbulence for Re=100 at Initial time step




RIKESH SHARMA (180606)
2D Turblence for Re=100 at n=2000 time step
ω (Vorticity)
Y
X
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

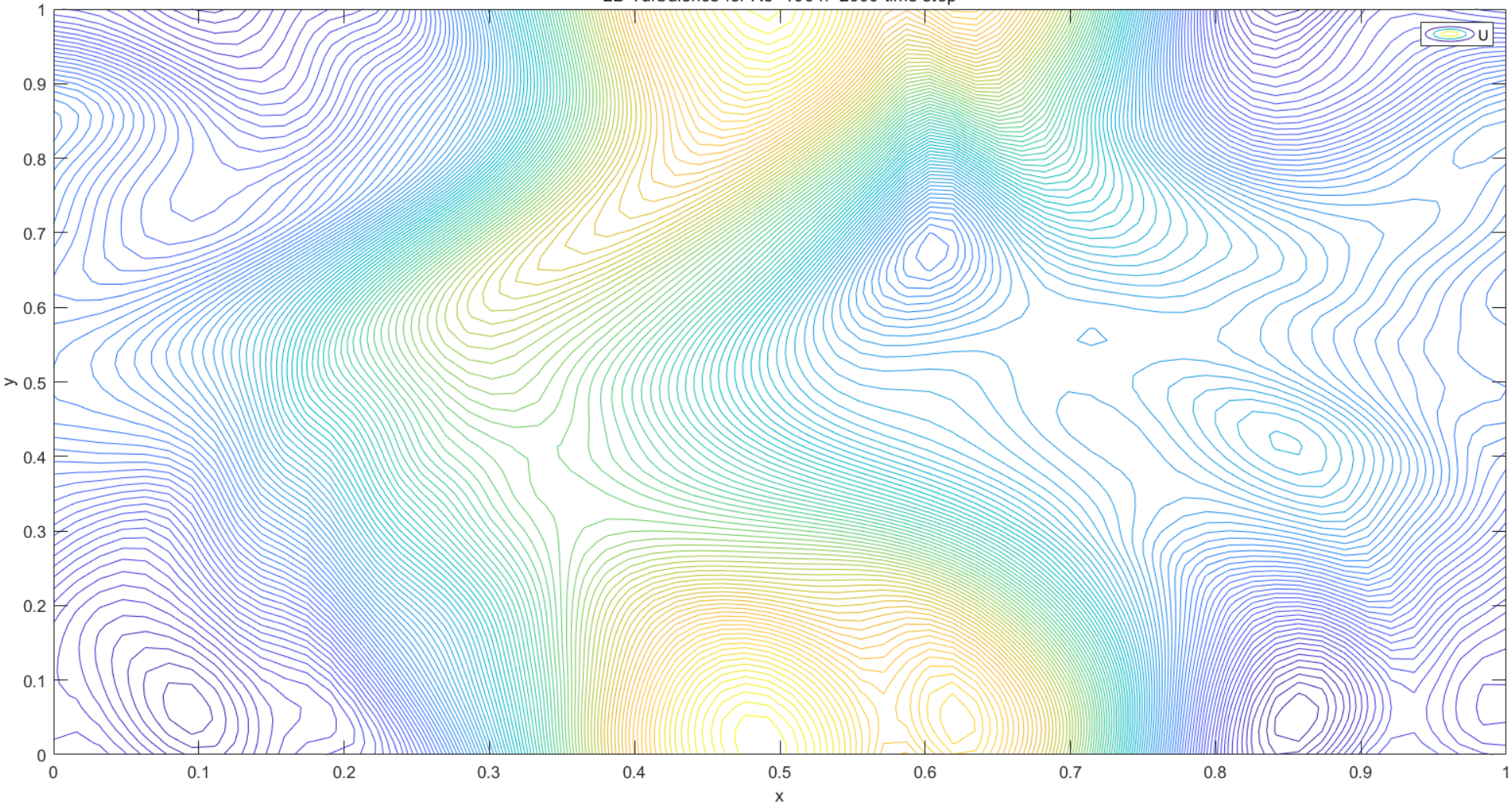
RIKESH SHARMA 180606

2D Turbulence for Re=100 n=2000 time step



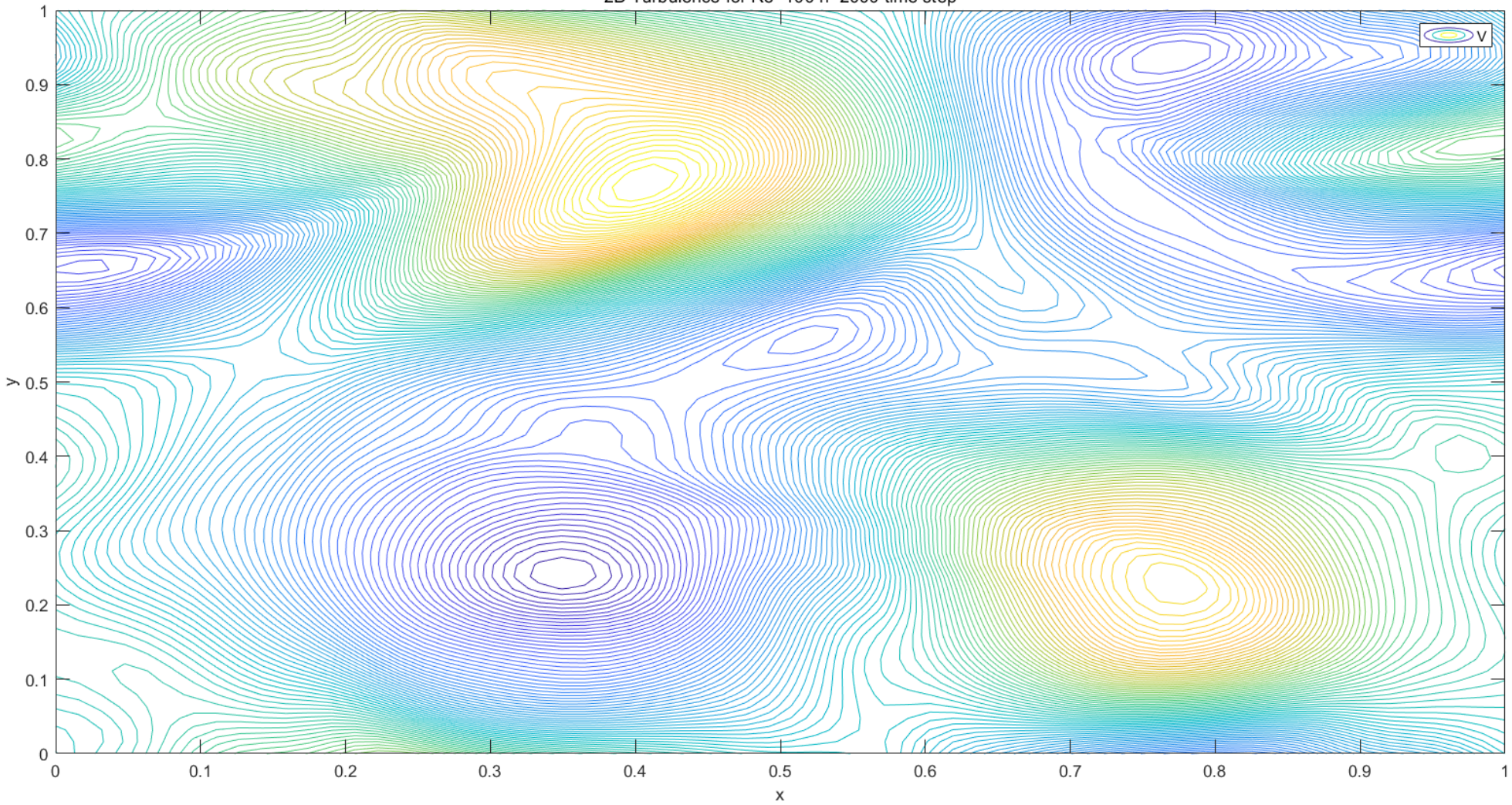
RIKESH SHARMA 180606

2D Turbulence for Re=100 n=2000 time step



RIKESH SHARMA 180606

2D Turbulence for Re=100 n=2000 time step



Discussions

1. Vorticity Transport equation is solved using RKW3 method and Stream (pressure poison equation type) Gauss seidel is used.
2. At initial time step Stream function is initialized with random values “rn” where -1 < rn < 1.
3. The random value of stream function injects a Turbulence in the domain.
4. Initial Turbulence characteristics can be seen in above figures, flow at this stage is highly turbulent as observed from figure.
5. As time passes this initial turbulence inject evolves and transmission of energy takes place from small scale to large scale, which can be seen from figures attached above for n=2000-time step.
6. It was also observed that there was dissipation of energy, due to viscous effects.
7. This dissipation has stabilizing effect on initial turbulence.
8. As Reynolds number increases the viscous effects is decreased hence it takes more time for flow to stabilize

Folder :

https://drive.google.com/drive/folders/1finpRfov_U7cGT7RVFJluGwzEwAisUgo?usp=sharing

Link to video file :

1. https://drive.google.com/file/d/1LBFDeCLElAhNetUPX0VKYsmUL3rGbvAX/view?usp=sharing
2. https://drive.google.com/file/d/1TULW7Z4qjngyLiMEMRfW_08-mQg2RKGJ/view?usp=sharing

Please find the MATLAB code (.m file) also included with the document

MATLAB code

```
%RIKESH SHARMA
%180606
```

```
%ME630A                submission on : 20/11/2021
function turbulence2D()
Lx=1;
Ly=1;
Nx=64;
Ny=64;
nxp2=Nx+2;
nyp2=Ny+2;
dx=(Lx)/(Nx-1);
dy=(Ly)/(Ny-1);
x=(0:dx:Lx);
y=(0:dy:Ly);
tmax=0.2;
dt=0.0001;
Nt=tmax/dt;
%since we have to initialize for psi with random number between -1
and 1
%over the whole domain we use a=-1 b=1 and use psi=a+(b-
a)*rand(nxp2,nyp2)
%to initialize psi with random number -1 and 1. Here rand(nxp2,nyp2)
%returns a matrix of size (nxp2 X nyp2) with random number between 0
to 1.
a=-1;  b=1;
psi=a+(b-a)*rand(nxp2,nyp2);
psi(Nx+2,:)=psi(2,:);
psi(1,:)=psi(Nx+1,:);
psi(:,Ny+2)=psi(:,2);
psi(:,1)=psi(:,Ny+1);
psi_old=psi;
%Initializing u0, v0, omega0 for t=0.0 using random value of psi over
% the domain
u=zeros(nxp2,nyp2);
u(2:Nx+1,2:Ny+1)=(psi(2:Nx+1,3:Ny+2)-psi(2:Nx+1,1:Ny))/(2*dy);
u(Nx+2,:)=u(2,:);
u(1,:)=u(Nx+1,:);
u(:,Ny+2)=u(:,2);
u(:,1)=u(:,Ny+1);

v=zeros(nxp2,nyp2);
v(2:Nx+1,2:Ny+1)=-(psi(3:Nx+2,2:Ny+1)-psi(1:Nx,2:Ny+1))/(2*dx);
v(Nx+2,:)=v(2,:);
v(1,:)=v(Nx+1,:);
v(:,Ny+2)=v(:,2);
v(:,1)=v(:,Ny+1);

omega=zeros(nxp2,nyp2);
omega(2:Nx+1,2:Ny+1)=(v(3:Nx+2,2:Ny+1)-v(1:Nx,2:Ny+1))/(2*dx) -
(u(2:Nx+1,3:Ny+2)-u(2:Nx+1,1:Ny))/(2*dy);
omega(Nx+2,:)=omega(2,:);
```

```
omega(1,:)=omega(Nx+1,:);
omega(:,Ny+2)=omega(:,2);
omega(:,1)=omega(:,Ny+1);

% [x,y]=meshgrid(x,y);
% contour(x,y,omega(2:Nx+1,2:Nx+1));
Re=100;
ctr=1;
[x,y]=meshgrid(x,y);
for n=1:Nt
    clf
    %RKW3
    omega_t=omega;
    K1= -u(2:Nx+1,2:Ny+1).*(omega_t(3:Nx+2,2:Ny+1)-
omega(1:Nx,2:Ny+1))/(2*dx)...
        -v(2:Nx+1,2:Ny+1).*(omega_t(2:Nx+1,3:Ny+2)-
omega(2:Nx+1,1:Ny))/(2*dy)...
        +1/Re*((omega_t(3:Nx+2,2:Ny+1)-
2*omega_t(2:Nx+1,2:Ny+1)+omega_t(1:Nx,2:Ny+1))/(dx*dx)...
        +      (omega_t(2:Nx+1,3:Ny+2)-
2*omega_t(2:Nx+1,2:Ny+1)+omega_t(2:Nx+1,1:Ny))/(dy*dy));

    omega_t(2:Nx+1,2:Ny+1)= omega_t(2:Nx+1,2:Ny+1) +dt/3*K1;
    K1=-5/9*K1   -u(2:Nx+1,2:Ny+1).*(omega_t(3:Nx+2,2:Ny+1)-
omega(1:Nx,2:Ny+1))/(2*dx)...
        -v(2:Nx+1,2:Ny+1).*(omega_t(2:Nx+1,3:Ny+2)-
omega(2:Nx+1,1:Ny))/(2*dy)...
        +1/Re*((omega_t(3:Nx+2,2:Ny+1)-
2*omega_t(2:Nx+1,2:Ny+1)+omega_t(1:Nx,2:Ny+1))/(dx*dx)...
        +      (omega_t(2:Nx+1,3:Ny+2)-
2*omega_t(2:Nx+1,2:Ny+1)+omega_t(2:Nx+1,1:Ny))/(dy*dy));

    omega_t(2:Nx+1,2:Ny+1) = omega_t(2:Nx+1,2:Ny+1) +15/16*dt*K1;
    K1 = -153/128 *K1  -u(2:Nx+1,2:Ny+1).*(omega_t(3:Nx+2,2:Ny+1)-
omega(1:Nx,2:Ny+1))/(2*dx)...
        -v(2:Nx+1,2:Ny+1).*(omega_t(2:Nx+1,3:Ny+2)-
omega(2:Nx+1,1:Ny))/(2*dy)...
        +1/Re*((omega_t(3:Nx+2,2:Ny+1)-
2*omega_t(2:Nx+1,2:Ny+1)+omega_t(1:Nx,2:Ny+1))/(dx*dx)...
        +      (omega_t(2:Nx+1,3:Ny+2)-
2*omega_t(2:Nx+1,2:Ny+1)+omega_t(2:Nx+1,1:Ny))/(dy*dy));

    omega(2:Nx+1,2:Ny+1)=omega_t(2:Nx+1,2:Ny+1)+ 8/15*dt*K1;
    omega(Nx+2,:)=omega(2,:);
    omega(1,:)=omega(Nx+1,:);
    omega(:,Ny+2)=omega(:,2);
    omega(:,1)=omega(:,Ny+1);
    %Gauss Siedel
    error=0.5;
```

```
    error_old=0;
    changeError=error-error_old;
    epsilon=1e-5;
    while (changeError>epsilon && ctr<10000)
        psi(Nx+2,:)=psi(2,:);
        psi(1,:)=psi(Nx+1,:);
        psi(:,Ny+2)=psi(:,2);
        psi(:,1)=psi(:,Ny+1);
        for i=2:Nx+1
            for j=2:Ny+1
                psi(i,j)=0.25*(psi_old(i+1,j)+psi_old(i,j+1)...
                                +psi(i-1,j)+psi(i,j-
1)+omega(i,j)*dx*dx);

            end
        end
%
psi(2:Nx+1,2:Ny+1)=0.25*(psi_old(3:Nx+2,2:Ny+1)+psi_old(2:Nx+1,3:Ny+2
)...
%
+psi(1:Nx,2:Ny+1)+psi(2:Nx+1,1:Ny)+omega(2:Nx+1,2:Ny+1)*dx*dx);
%          psi(Nx+2,:)=psi(2,:);
%          psi(1,:)=psi(Nx+1,:);
%          psi(:,Ny+2)=psi(:,2);
%          psi(:,1)=psi(:,Ny+1);

        error_old=error;
        errorMatrix=(psi(2:Nx+1,2:Ny+1)-psi_old(2:Nx+1,2:Ny+1));
        error = norm(errorMatrix,2);
        changeError=error-error_old;
        ctr=ctr+1;
        psi_old=psi;
    end
    %update BC
    omega(Nx+2,:)=omega(2,:);
    omega(1,:)=omega(Nx+1,:);
    omega(:,Ny+2)=omega(:,2);
    omega(:,1)=omega(:,Ny+1);
    psi(Nx+2,:)=psi(2,:);
    psi(1,:)=psi(Nx+1,:);
    psi(:,Ny+2)=psi(:,2);
    psi(:,1)=psi(:,Ny+1);

    %calculate u, v
    u(2:Nx+1,2:Ny+1)=(psi(2:Nx+1,3:Ny+2)-psi(2:Nx+1,1:Ny))/(2*dy);
    u(Nx+2,:)=u(2,:);
    u(1,:)=u(Nx+1,:);
    u(:,Ny+2)=u(:,2);
    u(:,1)=u(:,Ny+1);
```

```
    v(2:Nx+1,2:Ny+1)=-(psi(3:Nx+2,2:Ny+1)-psi(1:Nx,2:Ny+1))/(2*dx);
    v(Nx+2,:)=v(2,:);
    v(1,:)=v(Nx+1,:);
    v(:,Ny+2)=v(:,2);
    v(:,1)=v(:,Ny+1);
   % Save images Repeat
    contour(x,y,u(2:Nx+1,2:Nx+1),20);
     xlabel('x');
     ylabel('y');
     title('RIKESH SHARMA 180606','\omega 2D Turbulence for Re=100');
     movieFrame(n)=getframe;

end
myWriter=VideoWriter('turbulence2D');
myWriter.FrameRate = 30;
open(myWriter);
writeVideo(myWriter, movieFrame);
close(myWriter);

end
```

----------End of Document----------