The Simulations of the Particle Tracing by PYTHON

Reporter: Xing Wang

State Key Laboratory of Lunar and Planetary Sciences,

Space Science Institute

April 20, 2021

Contents

Question Description

Question 1 : Particle Tracing

An area of fluid ($1000 \times 1500\mathrm{m}$) is convecting with one central downwelling and two upwellings. The velocity field is given by the following equations:

$$v_x(x, y, t) = -v_{x,0} \sin\left[\frac{2\pi(x - c_x t)}{W}\right] \cos\left(\frac{\pi y}{H}\right) \tag{1}$$

$$v_y(x, y, t) = v_{y,0} \cos\left[\frac{2\pi(x - c_x t)}{W}\right] \sin\left(\frac{\pi y}{H}\right) \tag{2}$$

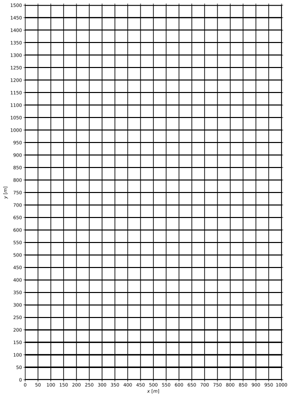
where x and y are respectively horizontal and vertical coordinates inside the box in m; $W = 1000\mathrm{m}$ and $H = 1500\mathrm{m}$ are the width and height of the model, respectively; $v_{x,0} = 1\mathrm{m/s}$ and $v_{y,0} = 3\mathrm{m/s}$ are scaling values for respectively horizontal and vertical velocity components; $c_x = 10\mathrm{m/s}$ is scaling value for advective velocity.

1. Visualise the velocity as an arrow field on a 2D grid of points 21×31. Show the movie with a time stepsize $\Delta t = 1\mathrm{s}$ till $t = 200\mathrm{s}$;
2. Lets assume that only velocities on the grids are provided. When $t = 0$, put a total number of 10000 particles uniformly distributed on the rectangular region $\{(x, y) | x \in (250, 750), y \in (0, 1500)\}$, then show the trajectories of the particles till $t = 200\mathrm{s}$.

The Visualization of Velocity Field

Q1: First step: making a 2D grid(21×31):

```python
# setting parameters
W=1000. # unit:m
H=1500. # unit:m
vx0=1. # unit:m/s
vy0=3. # unit:m/s
cx=10. # unit:m/s
dx=21  # unit: m
dy=31  # unit: m
dt=1   # unit: s
T=200  # unit: s
x=np.linspace(0,W,dx)
y=np.linspace(0,H,dy)
X,Y=np.meshgrid(x,y)
fig= plt.figure(figsize=(10,15))
ax = plt.gca()
for i in range(len(x)):
    plt.plot(X[:,i],Y[:,i],c='black')
    for j in range(len(y)):
        plt.plot(X[j,:],Y[j,:],c='black')
```

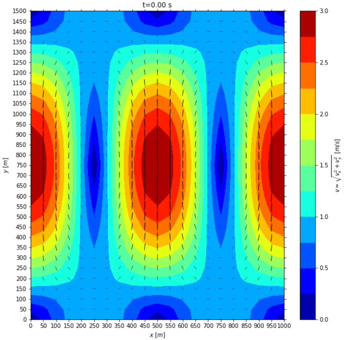


The Visualization of Velocity Field

Second step: Visualise the velocity as an arrow field:

```
x=np.linspace(0,W,dx)
y=np.linspace(0,H,dy)
X,Y=np.meshgrid(x,y)
for i in range(0,T,dt):
    vx=-vx0*np.sin(2*np.pi*(X-cx*dt*i)/W)*np.cos(np.pi*Y/H)
    vy=vy0*np.cos(2*np.pi*(X-cx*dt*i)/W)*np.sin(np.pi*Y/H)
    v=np.sqrt(vx**2+vy**2)
    fig= plt.figure(figsize=(10,15))
    ax = plt.gca()
    plt.contourf(X,Y,v,12, cmap=plt.get_cmap('jet'),interp='
                               bicubic')
    vel_clb=plt.colorbar()
    vel_clb.ax.set_ylabel('$v=\sqrt{v_{x}^{2}+v_{y}^{2}} \\
                               [m/s]$')
    Q=ax.quiver(X,Y,vx,vy,color='black',headlength=15,pivot=
                               'mid')
```

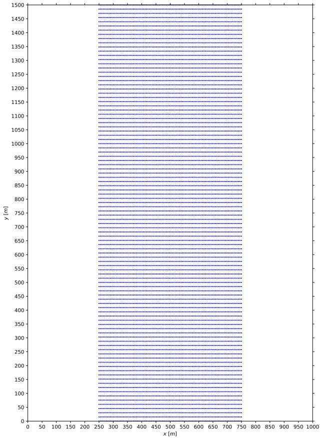
The result:


t=0.00 s
1500 1450 1400 1350 1300 1250 1200 1150 1100 1050 1000 950 900 850 800 750 700 650 600 550 500 450 400 350 300 250 200 150 100 50 0
y [m]
0 50 100 150 200 250 300 350 400 450 500 550 600 650 700 750 800 850 900 950 1000
x [m]
3.0 2.5 2.0 1.5 1.0 0.5 0.0
$v = \sqrt{v_x^2 + v_y^2}$ [m/s]

The Trajectories of Particles

Q2: First step: put a total number of 10000 particles:

```
x_num=100
y_num=100
num_data=x_num*y_num
x_box_inter=np.linspace(W/4,3*W/4,
                        x_num)
y_box_inter=np.linspace(0,H,y_num)
X_box_inter,Y_box_inter=np.meshgrid(
                        x_box_inter,
                        y_box_inter)
fig= plt.figure(figsize=(10,15))
ax = plt.gca()
plt.scatter(X_box_inter,Y_box_inter,
                        s=1,c='b')
```



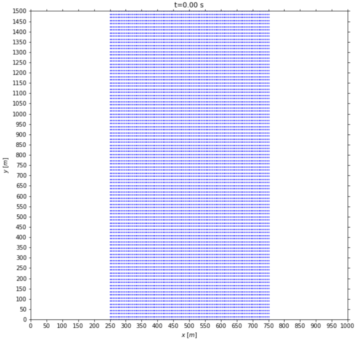
The Simulation of Particles Tracing

Second step: $\Delta t = t_{i+1} - t_i$, $r_x(x, y, t_{i+1}) = r_x(x, y, t_i) + v_x(x, y, t_i)\Delta t$, $r_y(x, y, t_{i+1}) = r_y(x, y, t_i) + v_y(x, y, t_i)\Delta t$.

```
delta_t=1
time=np.arange(0,T,delta_t)
pos_x=np.ones((len(x_box_inter),len(y_box_inter),len(time)))
pos_y=np.ones((len(x_box_inter),len(y_box_inter),len(time)))
vel_x=np.ones((len(x_box_inter),len(y_box_inter),len(time)))
vel_y=np.ones((len(x_box_inter),len(y_box_inter),len(time)))
pos_x[:,:,0]=X_box_inter
pos_y[:,:,0]=Y_box_inter
for k in range(len(time)-1):
    vx_k=-vx0*np.sin(2*np.pi*(X-cx*time[k])/W)*np.cos(np.pi*
                                        Y/H)
    vy_k=vy0*np.cos(2*np.pi*(X-cx*time[k])/W)*np.sin(np.pi*Y
                                        /H)
    f_vx_k_box = interpolate.interp2d(x,y,vx_k, kind='cubic'
                                        )
    f_vy_k_box = interpolate.interp2d(x,y,vy_k, kind='cubic'
                                        )
    vel_x[:,:,k]=f_vx_k_box(pos_x[0,:,k],pos_y[:,0,k])
    vel_y[:,:,k]=f_vy_k_box(pos_x[0,:,k],pos_y[:,0,k])
    pos_x[:,:,k+1]=pos_x[:,:,k]+vel_x[:,:,k]*delta_t
    pos_y[:,:,k+1]=pos_y[:,:,k]+vel_y[:,:,k]*delta_t
```

The Simulation of Particles Tracing

The result of simulation:



Thanks!