Lagrangian particle tracking experiment tutorial

Jang-Geun Choi

Center for Ocean Engineering, University of New Hampshire, NH, USA

Table of Contents

1. Governing equation

Particle tracking experiment is one of the simplest transport model but powerful tool that can be applicable to many problems. The governing equations is given by

$$\frac{d\vec{X}}{dt} = \vec{u}$$

In case of two-dimensional problem, $\vec{X} = (X, Y)$ and $\vec{u} = (u, v)$. Note that equation above is a system of equations that consists of two ordinary differential equations ($dX/dt = u$ and $dY/dt = v$). It is not hard to solve the system of equation using numerical approach. Based on first order forward Euler scheme, the equations can be discretized to

$$\frac{X^{i+1} - X^i}{\Delta t} = u^i$$

$$\frac{Y^{i+1} - Y^i}{\Delta t} = v^i$$

so

$$X^{i+1} = X^i + \Delta t\, u^i$$

$$Y^{i+1} = Y^i + \Delta t\, v^i$$

where upper script i and $i+1$ represents variables in current and nex step in time, respectively. Therefore, based on given initial position of position (X^0 and Y^0) and given velocity fields (u and v), position in first step ($t = \Delta t$) can be calculated, and then that of second step ($t = 2\Delta t$) can be calculated using the previous step. This can be continued.

2. Practice for model development and numerical error

Consider very simple velocity fields given by

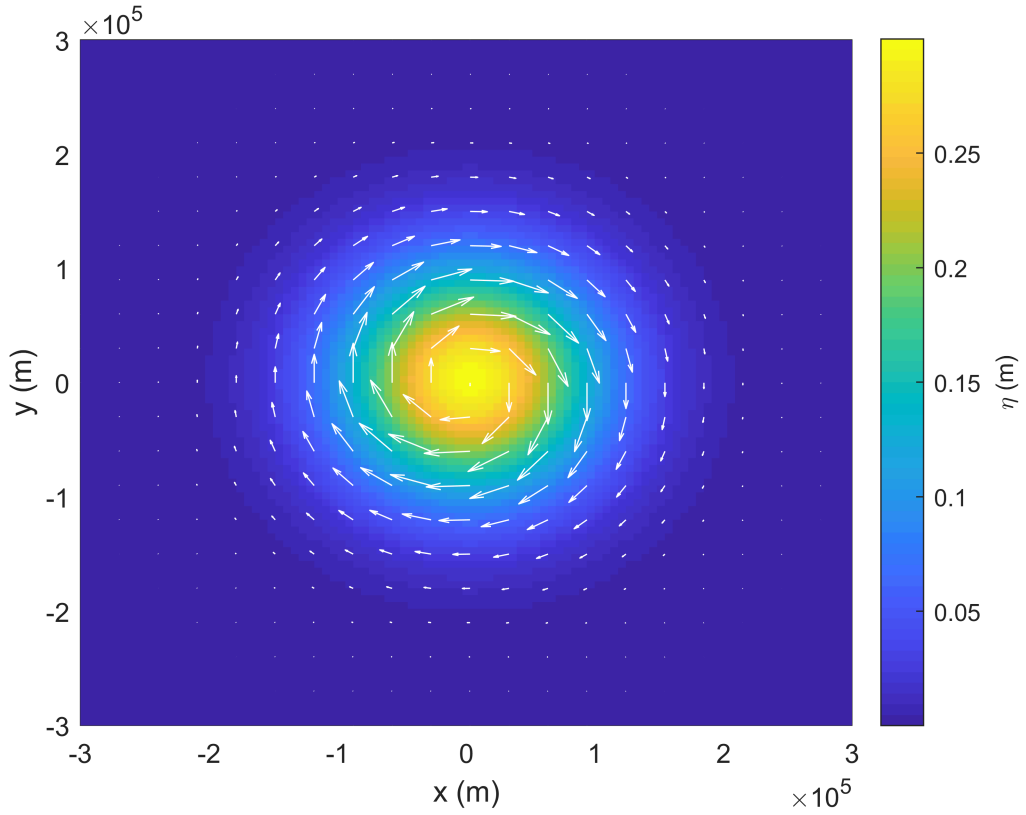
$$\eta = \eta_0 e^{-(x^2+y^2)/L^2}$$

$$v = \frac{g}{f}\frac{\partial \eta}{\partial x} = -\frac{g\eta_0}{f}\frac{2x}{L^2}e^{-(x^2+y^2)/L^2}$$

$$u = -\frac{g}{f}\frac{\partial \eta}{\partial y} = \frac{g\eta_0}{f}\frac{2y}{L^2}e^{-(x^2+y^2)/L^2}$$

where $\eta_0 = 0.3\,m$, $g = 10\,m\,s^{-2}$, $f = 10^{-4}\,s^{-1}$, $L = 10^5\,m$. This analytical equations describes simple clockwise eddy governed by pure geostrophic current.

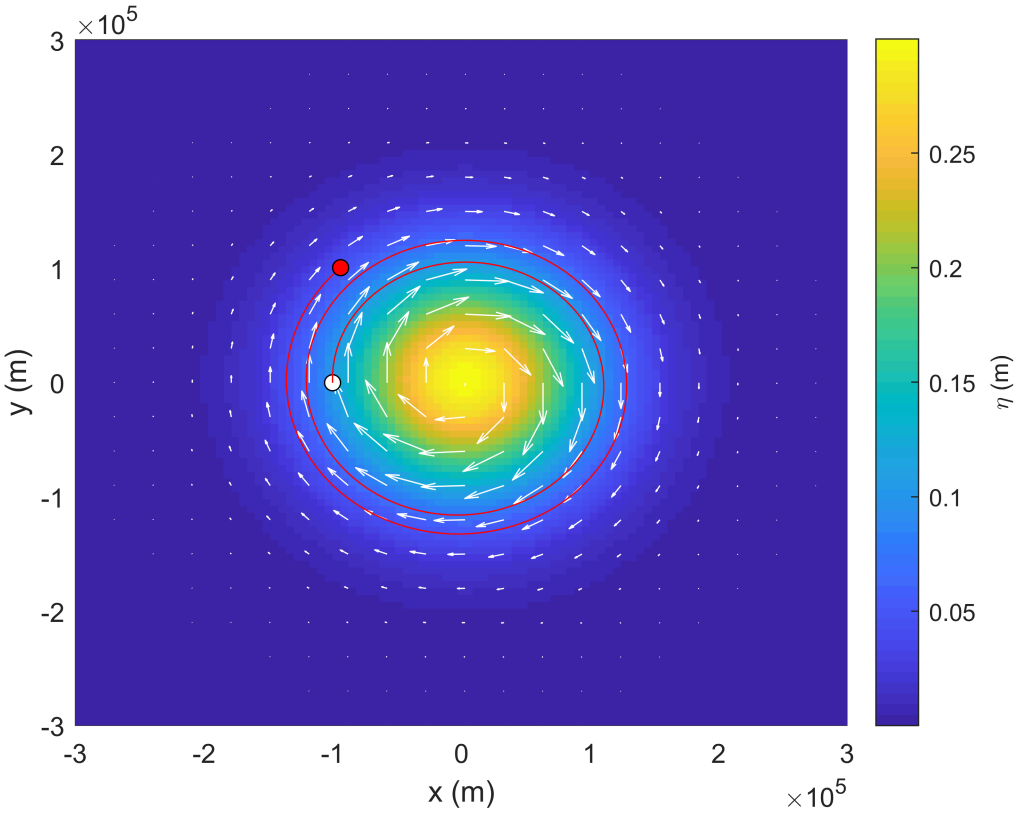
```
clc;clear;close all;

eta0=0.3;
g=10;
f=1e-4;
L=1e5;

x1=linspace(-3*L,3*L,100);
y1=linspace(-3*L,3*L,101);
[y,x]=meshgrid(y1,x1);

eta=eta0*exp(-(x.^2+y.^2)/L^2);
v=-g/f*2*x/L^2.*exp(-(x.^2+y.^2)/L^2);
u=g/f*2*y/L^2.*exp(-(x.^2+y.^2)/L^2);

pcolor(x,y,eta)
hold on
nn=5;
quiver(x(1:nn:end,1:nn:end),y(1:nn:end,1:nn:end),...
    u(1:nn:end,1:nn:end),v(1:nn:end,1:nn:end),'w')
shading flat
xlabel('x (m)')
ylabel('y (m)')
cb=colorbar;
ylabel(cb,'\eta (m)')
```



As initial condition (location) of particle, consider $X^0 = -1 \times 10^5 m$ and $Y^0 = 0\,m$. Position at the next step can be calculated by $X^{i+1} = X^i + \Delta t\, u^i$ and $Y^{i+1} = Y^i + \Delta t\, v^i$ mentioned above where time step will be sett to $\Delta t = 10^3 s$ and 100 steps will be calculated. It is worth noting that u^i and v^i can be interpolated from u and v.

```
dt=1e4;
X0=-1e5;
Y0=0;

n=3000000/dt;
T=(0:dt:n*dt)';
X=NaN(size(T));
Y=NaN(size(T));

X(1)=X0;
Y(1)=Y0;

for i=1:n
    ui=interp2(y,x,u,Y(i),X(i));
    vi=interp2(y,x,v,Y(i),X(i));

    X(i+1)=X(i)+ui*dt;
    Y(i+1)=Y(i)+vi*dt;
end
```

```
hold on
plot(X0,Y0,'ok','MarkerFaceColor','w')
plot(X,Y,'r')
plot(X(end),Y(end),'ok','MarkerFaceColor','r')
```



Note that the particle slowly move down and η experienced by the particle decrease as time goes. This is numerical error. To be specific, the equations for velocity field satisfying

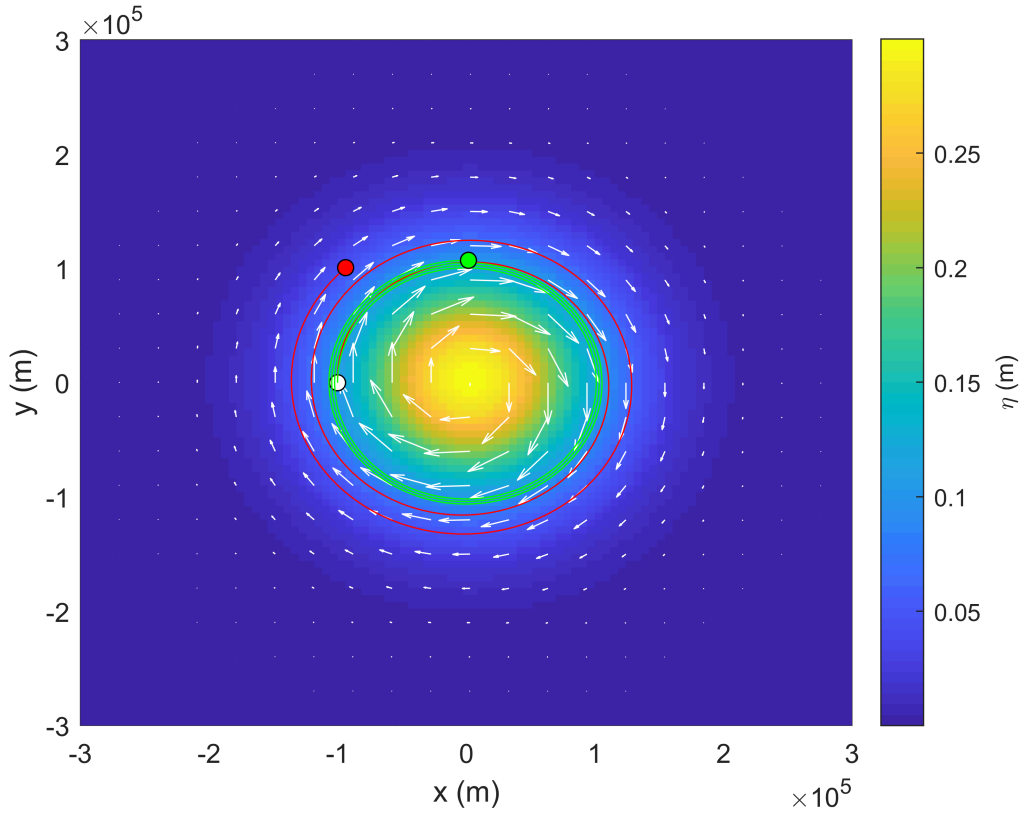
$$\frac{\partial \eta}{\partial t} + u\frac{\partial \eta}{\partial x} + v\frac{\partial \eta}{\partial y} = 0 \leftrightarrow \frac{D\eta}{Dt} = 0.$$

It can be mathematically verified by substituting the equations (definitions) for η, u, and v into the equation above. The equation above means that value of η should not be changed as time in terms of Lagrangian point of view. Therefore, analytically, particle follows contour-line of η to conserve the value.

The numerical error can be reduced by choosing smaller time step.

```
clear X Y

dt=1e3;

n=3000000/dt;
T=(0:dt:n*dt)';
X=NaN(size(T));
Y=NaN(size(T));
```

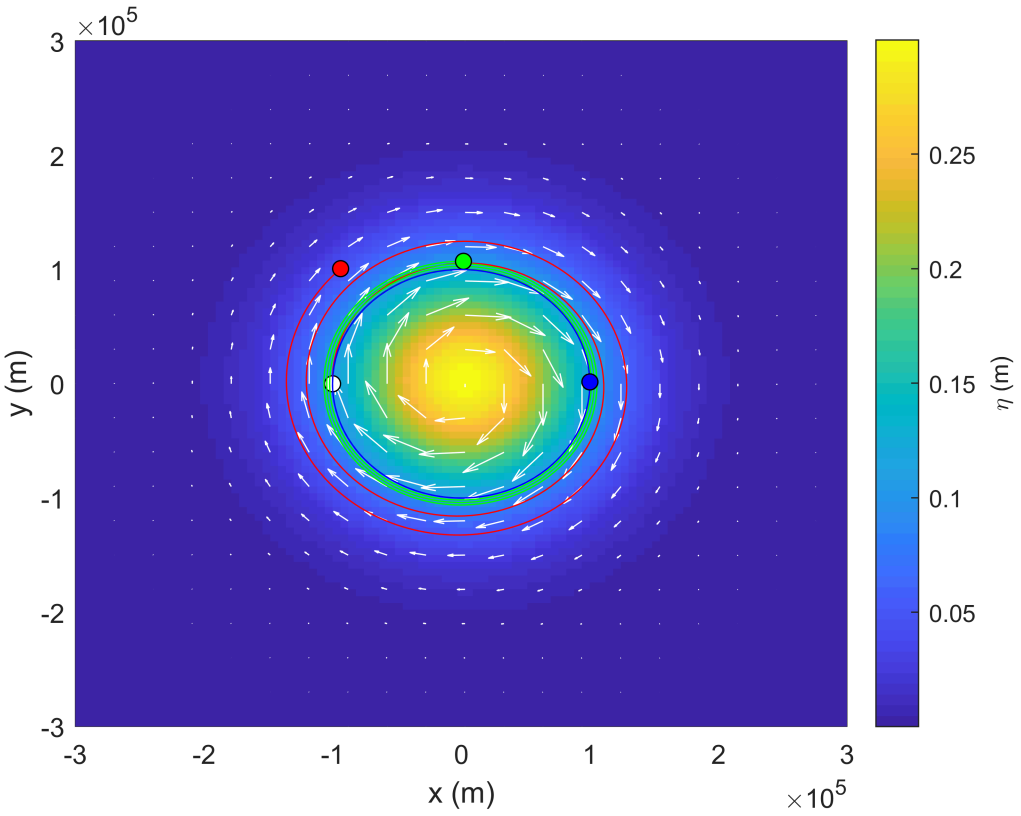
```
X(1)=X0;
Y(1)=Y0;

for i=1:n
    ui=interp2(y,x,u,Y(i),X(i));
    vi=interp2(y,x,v,Y(i),X(i));

    X(i+1)=X(i)+ui*dt;
    Y(i+1)=Y(i)+vi*dt;
end

plot(X,Y,'g')
plot(X(end),Y(end),'ok','MarkerFaceColor','g')
```



Another approach is to use better (higher order) numerical scheme.

```
clear X Y

dt=1e4;

n=3000000/dt;
T=(0:dt:n*dt)';
X=NaN(size(T));
Y=NaN(size(T));

X(1)=X0;
```

```
Y(1)=Y0;

for i=1:n
    ui1=interp2(y,x,u,Y(i),X(i));
    vi1=interp2(y,x,v,Y(i),X(i));

    Xi=X(i)+ui1*dt;
    Yi=Y(i)+vi1*dt;

    ui2=interp2(y,x,u,Yi,Xi);
    vi2=interp2(y,x,v,Yi,Xi);

    X(i+1)=X(i)+(ui1+ui2)/2*dt;
    Y(i+1)=Y(i)+(vi1+vi2)/2*dt;
end

plot(X,Y,'b')
plot(X(end),Y(end),'ok','MarkerFaceColor','b')
```

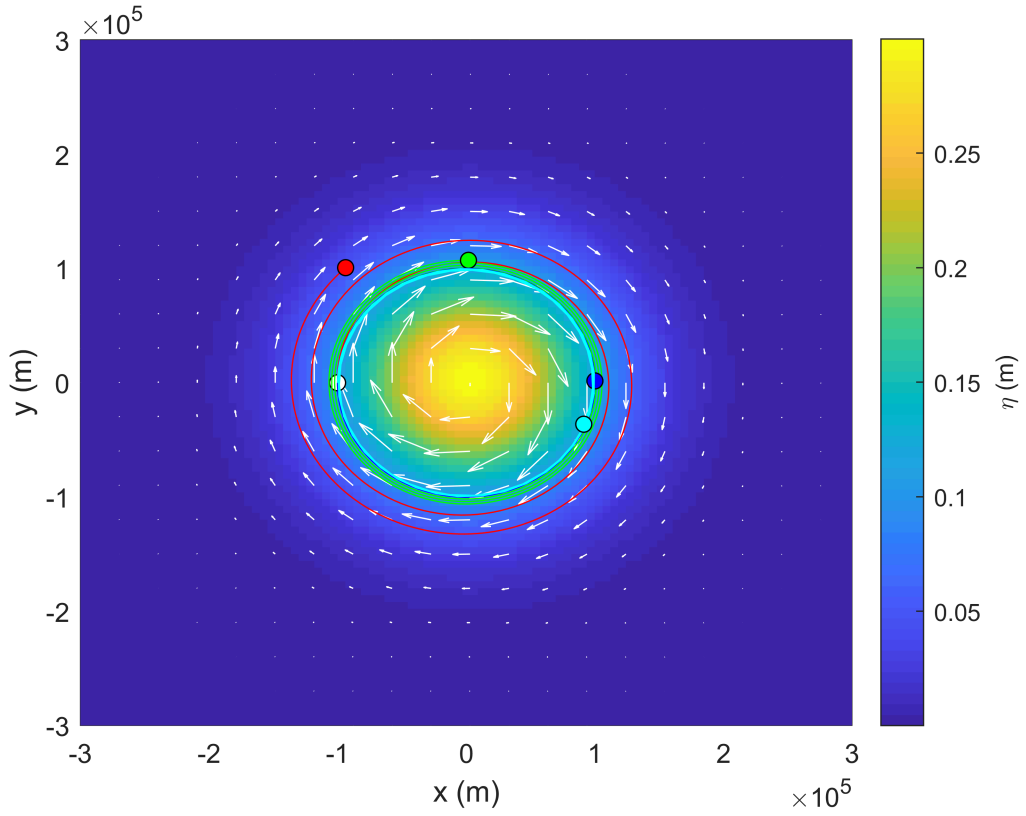


Note that this scheme has better accuracy (much more conservative η) regardless of 10 times longer time step.

MATLAB provides various ordinary equation solvers (https://www.mathworks.com/help/matlab/ordinary-differential-equations.html). Runge-Kutta scheme is one of the most famous solver and frequently adopted to particle tracking experiments (Van Sebille et al., 2018). Below is an example code for application of the

scheme using MATLAB built-in function. It is worth noting that a guidance for choosing solver is provided by MathWorks (https://www.mathworks.com/help/matlab/math/choose-an-ode-solver.html).

```
clear X Y

X(1)=X0;
Y(1)=Y0;

[t,XY]=ode45(@(t,Xv)uv(t,Xv,x,y,u,v),[0 3000000],[X Y]);
X=XY(:,1);
Y=XY(:,2);

plot(X,Y,'c')
plot(X(end),Y(end),'ok','MarkerFaceColor','c')
```



It is worth noting that accuracy of particle tracking experiments depends on not only accuracy of temporal integration (choice of solver) but also that of velocity interpolation. Ocassionally, special interpolation methods are adopted (Döös et al., 2013;Van Sebille et al., 2018; Delandmeter and Van Sebille, 2019; https://www.youtube.com/watch?v=QKSbbi_67Ro&list=PLKUjU2F1eF1jRT2mWrRGDlLd58Kjb6-er&index=3&t=1497s).

3. Diffusion and random-walk

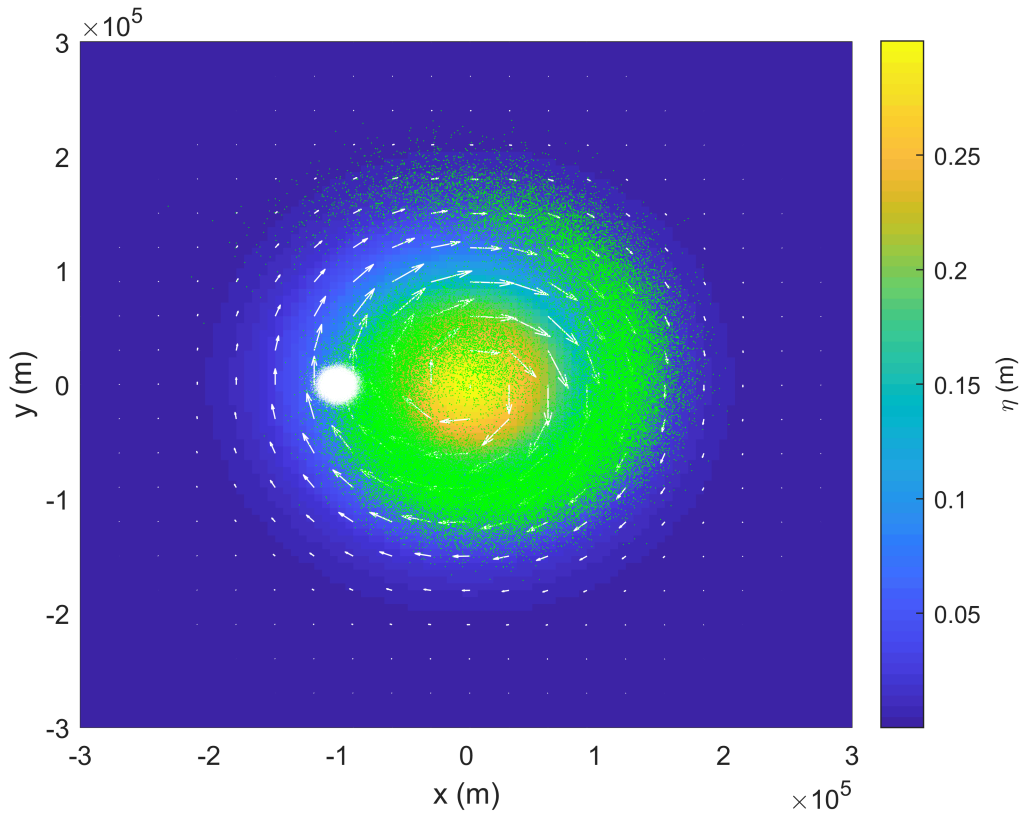
Subjects of transport is not only advected but also diffused. Diffusion process in lagrangian point of view can be considered as random-walk. The governing equation considering diffusion as random-walk is given by

$$\frac{d\vec{X}}{\mathrm{dt}} = \vec{u} + N(0, 2\Delta \mathrm{tA})$$

where $N(\mu, \sigma)$ represents random number from normal distribution of which mean μ and standard deviation σ. Below is example code for particle tracking experiments generalized to launch large number of particle at once and to consider diffusion via random-walk.

```
clc;clear;

eta0=0.3;
g=10;
f=1e-4;
L=1e5;

x1=linspace(-3*L,3*L,100);
y1=linspace(-3*L,3*L,101);
[y,x]=meshgrid(y1,x1);

eta=eta0*exp(-(x.^2+y.^2)/L^2);
v=-g/f*2*x/L^2.*exp(-(x.^2+y.^2)/L^2);
u=g/f*2*y/L^2.*exp(-(x.^2+y.^2)/L^2);

dt=1e4;
np=1e5; % number of particles
A=1e3;  % diffusion coefficient (m^2/s)
X0=-1e5;
Y0=0;
L0=6e3;

n=600000/dt;
T=(0:dt:n*dt)';
X=NaN(length(T),np);
Y=NaN(length(T),np);

X(1,:)=normrnd(X0,L0,1,np);
Y(1,:)=normrnd(Y0,L0,1,np);

tic
for i=1:n
 ui1=interp2(y,x,u,Y(i,:),X(i,:));
 vi1=interp2(y,x,v,Y(i,:),X(i,:));

 Xi=X(i,:)+ui1*dt;
 Yi=Y(i,:)+vi1*dt;

 ui2=interp2(y,x,u,Yi,Xi);
 vi2=interp2(y,x,v,Yi,Xi);

 X(i+1,:)=X(i,:)+(ui1+ui2)/2*dt+normrnd(0,sqrt(2*dt*A),1,np);
 Y(i+1,:)=Y(i,:)+(vi1+vi2)/2*dt+normrnd(0,sqrt(2*dt*A),1,np);
end
T_lagrangian=toc;
```

```
figure
pcolor(x,y,eta)
hold on
nn=5;
quiver(x(1:nn:end,1:nn:end),y(1:nn:end,1:nn:end),...
 u(1:nn:end,1:nn:end),v(1:nn:end,1:nn:end),'w')
shading flat
xlabel('x (m)')
ylabel('y (m)')
cb=colorbar;
ylabel(cb,'\eta (m)')

clr=lines(2);
plot(X(end,:),Y(end,:),'.','Color','g','MarkerSize',1);
plot(X(1,:),Y(1,:),'.','Color','w','MarkerSize',0.1);
hold off
```



It is worth noting that the equation above can be rewritten as

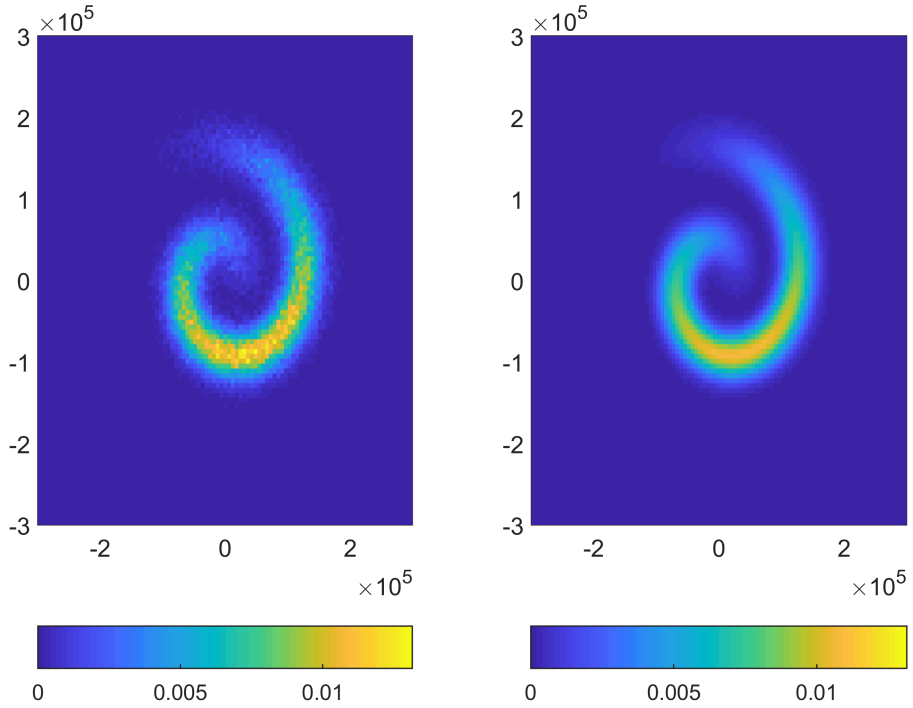
$$\frac{\partial C}{\partial t} + u\frac{\partial C}{\partial x} + v\frac{\partial C}{\partial y} = A\left(\frac{\partial^2 C}{\partial x^2} + \frac{\partial^2 C}{\partial y^2}\right)$$

in terms of continous Eulerian point of view. Therefore, the Lagrangian particle tracking experiments is not different with Eulerian tracer (ink) experiments. Below code is for Eulerian ink experiments.

```
C0=hist3([X(1,:)' Y(1,:)'],'Ctrs',{x1 y1});
Cmax=max(C0(:));
```

```
C=hist3([X(end,:)' Y(end,:)'],'Ctrs',{x1 y1})/Cmax;

figure
subplot(1,2,1)
pcolor(x,y,C)
shading flat
colorbar('southoutside')
cx=caxis;

clearvars -except cx Cmax A T_lagrangian

global dx dy dt nx ny

eta0=0.3;
g=10;
f=1e-4;
L=1e5;

x1=linspace(-3*L,3*L,100);
y1=linspace(-3*L,3*L,101);
[y,x]=meshgrid(y1,x1);

xu1=conv(x1,[0.5 0.5],'valid');
yu1=y1;
xv1=x1;
yv1=conv(y1,[0.5 0.5],'valid');

[yu,xu]=meshgrid(yu1,xu1);
[yv,xv]=meshgrid(yv1,xv1);

eta=eta0*exp(-(x.^2+y.^2)/L^2);
v=-g/f*2*xv/L^2.*exp(-(xv.^2+yv.^2)/L^2);
u=g/f*2*yu/L^2.*exp(-(xu.^2+yu.^2)/L^2);

dx=mean(diff(x1));
dy=mean(diff(y1));
dt=1e3;
nx=size(x,1);
ny=size(y,2);

X0=-1e5;
Y0=0;
L0=6e3;
n=600000/dt;

c=zeros(nx,ny,n);
c(:,:,1)=exp(-(((x-X0)/L0).^2+((y-Y0)/L0).^2)/2);

tic
for i=1:n
    % advection
    c(:,:,i+1)=c(:,:,i)-dt*Adv(u,v,c(:,:,i));
    % diffusion
```

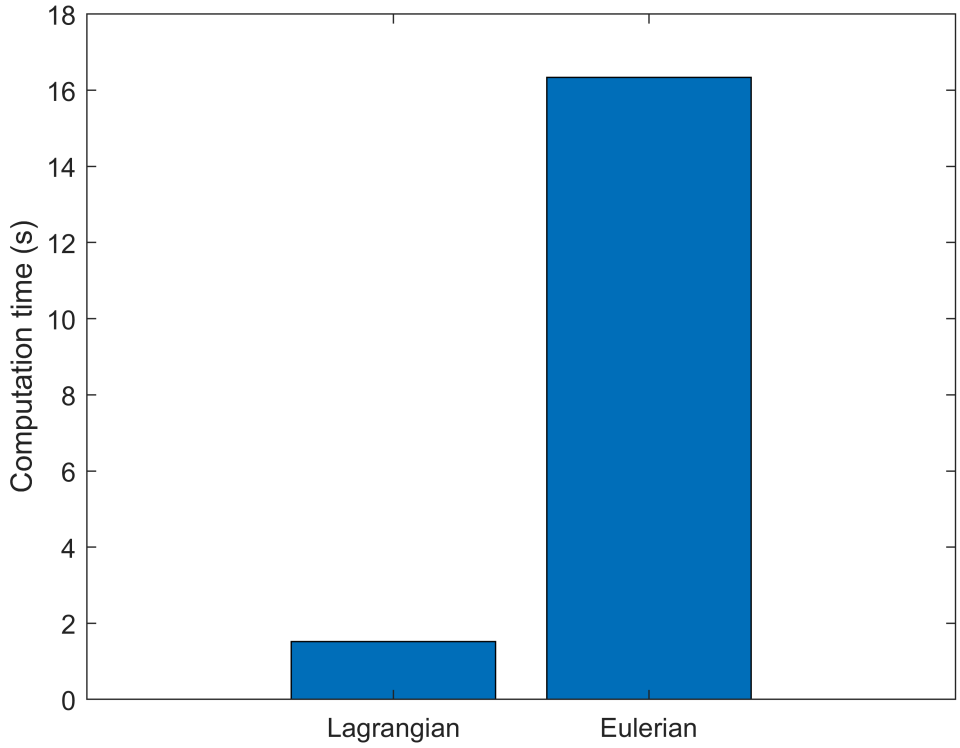
```
    c(2:end-1,2:end-1,i+1)=c(2:end-1,2:end-1,i+1)+...
        A*dt/dx^2*convn(c(:,:,i+1),[0 1 0;1 -4 1;0 1 0],'valid');

    c(:,1,i+1)=0;
    c(:,end,i+1)=0;
    c(1,:,i+1)=0;
    c(end,:,i+1)=0;

end
T_eulerian=toc;

subplot(1,2,2)
pcolor(x,y,c(:,:,end))
shading flat
colorbar('southoutside')
caxis(cx)
```



Left figure indicate normalized histogram of particles by maximum initial number of particle in a bin. Right figure is Eulerian ink experiment result using consistent parameters with the Lagrangian particle experiment. Both are almost identical and consistent. Note that computation time of Lagrangian particle tracking is notably faster than that of Eulerian ink experiment, that is one of the reason for popularity of Lagrangian particle tracking approaches. Figure below is highly possibly exaggerated because the code for the Eulerian ink here is not well written in terms of MATLAB runtime efficienency.

```
figure
bar([T_lagrangian T_eulerian])
```

```
set(gca,'XTickLabel',{'Lagrangian','Eulerian'})
ylabel('Computation time (s)')
```




```
function dXdt = uv(~,X,x,y,u,v)
dXdt(1) = interp2(y,x,u,X(2),X(1)); %dX/dt=u
dXdt(2) = interp2(y,x,v,X(2),X(1)); %dY/dt=v
dXdt=dXdt';
end
```

Reference

Döös, K., Kjellsson, J., & Jönsson, B. (2013). TRACMASS—A Lagrangian trajectory model. *Preventive methods for coastal protection: Towards the use of ocean dynamics for pollution control*, 225-249.

Van Sebille, E., Griffies, S. M., Abernathey, R., Adams, T. P., Berloff, P., Biastoch, A., ... & Zika, J. D. (2018). Lagrangian ocean analysis: Fundamentals and practices. *Ocean modelling*, *121*, 49-75.

Delandmeter, P., & Van Sebille, E. (2019). The Parcels v2. 0 Lagrangian framework: new field interpolation schemes. *Geoscientific Model Development*, *12*(8), 3571-3584.