Vortex Leapfrogging

1 Introduction

Vortex interactions are an area of fluid dynamics still undergoing significant research. In this piece of work, we will first explore the dynamics of vortices and their interactions, then analyse the stability of a system of four vortices, following the work by Acheson (2000). A program was written in MATLAB to plot the paths of vortices in this system, and this is given in Appendix 1.

1.1 Notation

Many differing notations are used in the literature for differential operators, vorticity, vectors and other terms. In this piece of work, the first partial derivative is represented by

$$\frac{\partial}{\partial t} = \partial_t, \tag{1.1}$$

and the second by

$$\frac{\partial^2}{\partial t^2} := \partial_{tt}, \tag{1.2}$$

$$\frac{\partial^2}{\partial x \partial y} := \partial_{xy} \tag{1.3}$$

for double and mixed derivatives. Vectors are written

$$\boldsymbol{a} = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix}, \tag{1.4}$$

and velocity is written (in three dimensional cartesian coordinates) as

$$\boldsymbol{u} = \begin{pmatrix} u_x \\ u_y \\ u_z \end{pmatrix}. \tag{1.5}$$

1.2 Vector Identities

The following vector identities are used in this piece of work. There is much use of the so-called "Del" operator, represented by (for Cartesian coordinates)

$$\nabla = \begin{pmatrix} \partial_x \\ \partial_y \\ \partial_z \end{pmatrix}. \tag{1.6}$$

In this notation, the divergence of a fluid is given by the inner product of "Del" with the velocity

$$\mathrm{div}(\boldsymbol{u}) = \nabla \cdot \boldsymbol{u} = \partial_x u_x + \partial_y u_y + \partial_z u_z, \tag{1.7}$$

and the curl is given by the outer product

$$\mathrm{curl}(\boldsymbol{u}) = \nabla \times \boldsymbol{u} = \begin{pmatrix} \partial_y u_z - \partial_z u_y \\ \partial_z u_x - \partial_x u_z \\ \partial_x u_y - \partial_y u_x \end{pmatrix}. \tag{1.8}$$

The triple outer product of $\boldsymbol{u}$ and ∇ can be written using the following identity (from Harlen (2014))

$$\boldsymbol{u} \times (\nabla \times \boldsymbol{u}) = \frac{1}{2}\nabla(\boldsymbol{u}^2) - \boldsymbol{u} \cdot \nabla \boldsymbol{u}. \tag{1.9}$$

2 Dynamics of Vortices

The general motion of a fluid is given by the Navier-Stokes equations, which are usually written in vector form:

$$\partial_t \boldsymbol{u} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = \nu \nabla^2 \boldsymbol{u} - \frac{1}{\rho}\nabla P, \tag{2.1}$$

where ρ is the fluid density, P is the pressure and ν is the *kinematic viscosity*, μ/ρ. When working with incompressible fluids, we also use the continuity equation

$$\nabla \cdot \boldsymbol{u} = 0. \tag{2.2}$$

This can be reworked to find an equation for the local vorticity of the fluid by remembering that the vorticity $\boldsymbol{\omega}$ is equivalent to the vector curl of the velocity $\boldsymbol{u}$. Following Harlen (2014), we can rewrite eq. (2.1) using the identity in eq. (1.9)

$$\partial_t \boldsymbol{u} + \frac{1}{2}\nabla(\boldsymbol{u}^2) - \boldsymbol{u} \times (\nabla \times \boldsymbol{u}) = \nu \nabla^2 \boldsymbol{u} - \frac{1}{\rho}\nabla P. \tag{2.3}$$

Taking the curl of the whole equation, we obtain

$$\partial_t \nabla \times \boldsymbol{u} + \frac{1}{2}\nabla \times (\nabla(\boldsymbol{u}^2)) - \nabla \times (\boldsymbol{u} \times (\nabla \times \boldsymbol{u})) = \nabla \times (\nu \nabla^2 \boldsymbol{u}) - \nabla \times (\frac{1}{\rho}\nabla P). \quad (2.4)$$

Now rewriting in terms of $\boldsymbol{\omega}$ and noting that the outer product of ∇ with a gradient is zero always, many parts of eq. (2.4) disappear and we are left with

$$\partial_t \boldsymbol{\omega} - \nabla \times (\boldsymbol{u} \times \boldsymbol{\omega}) = \nu \nabla^2 \boldsymbol{\omega}. \quad (2.5)$$

Using another expanded version of the triple vector product (again following Harlen (2014))

$$\nabla \times (\boldsymbol{u} \times \boldsymbol{\omega}) = \boldsymbol{u}\nabla \cdot \boldsymbol{\omega} + \boldsymbol{\omega} \cdot \nabla \boldsymbol{u} - \boldsymbol{\omega}\nabla \cdot \boldsymbol{u} - \boldsymbol{u} \cdot \nabla \boldsymbol{\omega}, \quad (2.6)$$

we can then write

$$\partial_t \boldsymbol{\omega} - \boldsymbol{u}\nabla \cdot \boldsymbol{\omega} + \boldsymbol{\omega} \cdot \nabla \boldsymbol{u} - \boldsymbol{\omega}\nabla \cdot \boldsymbol{u} - \boldsymbol{u} \cdot \nabla \boldsymbol{\omega} = \nu \nabla^2 \boldsymbol{\omega}. \quad (2.7)$$

However, from the continuity equation (eq. (2.2)) we have that $\nabla \cdot \boldsymbol{u} = 0$, and we also know that the divergence of a curl is always zero, so as $\boldsymbol{\omega} = \nabla \times \boldsymbol{u}$, $\nabla \cdot \boldsymbol{\omega} = 0$. This means we can further rewrite eq. (2.7) to obtain

$$\partial_t \boldsymbol{\omega} + \boldsymbol{u} \cdot \nabla \boldsymbol{\omega} = \boldsymbol{\omega} \cdot \nabla \boldsymbol{u} + \nu \nabla^2 \boldsymbol{\omega}, \quad (2.8)$$

the *vorticity equation.*

Appendix I: MATLAB Code

MATLAB code for the vortex leapfrogging program is given below.

```
clear all
N=input('number of vortices = ');
%set time step and scales
t=0;
T=input('number of time steps = ');
%set length step and scales
h=input('step-size = ');
%define vortex strengths
for i=1:N
    k(i)=input(['strength of vortex ',num2str(i),' = ']);
end
%define variable for position
x=NaN(1000,N);
```

```
y=NaN(1000,N);
for i=1:N
    x(1,i)=input(['x-coordinate of vortex ',num2str(i),' = ']);
    y(1,i)=input(['y-coordinate of vortex ',num2str(i),' = ']);
end
for a=2:T
    clear x1 x2 x3 x4 y1 y2 y3 y4 k1 k2 k3 k4 l1 l2 l3 l4
    x1=x(a-1,:);
    y1=y(a-1,:);
    k1=NaN(1,N);
    k2=NaN(1,N);
    k3=NaN(1,N);
    k4=NaN(1,N);
    l1=NaN(1,N);
    l2=NaN(1,N);
    l3=NaN(1,N);
    l4=NaN(1,N);
    for i=1:N
        k1(i)=h*vortexf(i,k,x1,y1,N);
        l1(i)=h*vortexg(i,k,x1,y1,N);
    end
    x2=x1+(k1/2);
    y2=y1+(y1/2);
    for i=1:N
        k2(i)=h*vortexf(i,k,x2,y2,N);
        l2(i)=h*vortexg(i,k,x2,y2,N);
    end
    x3=x1+(k2/2);
    y3=y1+(y2/2);
    for i=1:N
        k3(i)=h*vortexf(i,k,x3,y3,N);
        l3(i)=h*vortexg(i,k,x3,y3,N);
    end
    x4=x1+(k3);
    y4=y1+(y3);
    for i=1:N
        k4(i)=h*vortexf(i,k,x4,y4,N);
        l4(i)=h*vortexg(i,k,x4,y4,N);
    end
    x(a,:)=x(a-1,:)+((1/6)*(k1+2*k2+2*k3+k4));
    y(a,:)=y(a-1,:)+((1/6)*(l1+2*l2+2*l3+l4));
    percent=100*a/T;
```

```
    display([num2str(percent),'\% done'])
end
clf
multicomet(x,y)
title([num2str(T),' time steps, step size = ',num2str(h)])
```

The scheme used is a fourth order Runge-Kutta method with non-adaptive step size. The equations plotted are from Acheson (2000).

References

Acheson, D. (2000). Instability of Vortex Leapfrogging, *European Journal of Physics* **21**: 269–273.

Harlen, O. (2014). MATH3620 chapter 6, [online]. [Accessed 14/02/15] Available from http://vlebb.leeds.ac.uk.