Characteristic Mapping Method for Incompressible Euler Equations

Badal Yadav

M.Eng. Non-Thesis

Department of Mechanical Engineering

McGill University

Supervisor: Prof. Jean-Christophe Nave

Contents

Abstract

The characteristic mapping method [12] is a method for solving linear advection problems with arbitrary initial conditions. Its unique property is the decoupling of the computational and solution representation grids, thus allowing small length scales to be accurately represented in the solution with overall low computational cost. In this work, the idea is extended to non-linear problems thus requiring a more involved approach. Starting with applying the characteristic mapping method to model one-dimensional non-linear problems, we study the key properties of the method. Thus we extend this to solve the incompressible Euler equations in two-dimensions. We present several simulations of two-dimensional Euler equations for the cases of Kelvin-Helmholtz instability and vortex-merging. Finally, we investigate the multi-scale property of our approach.

Chapter 1

Introduction

The governing equations in fluid dynamics are a set of conservation laws - the continuity, momentum and energy equations. Most crucial of all is the momentum equation, which in its full form is known as the Navier-Stokes equation. Different regimes of fluid dynamics involve the study of these equations under different assumptions. In the scope of this report, we concern ourselves with incompressible, adiabatic and inviscid flows. The governing equation in this case simplifies to the incompressible Euler equations. We use this system as a problem to be solved by extending the characteristic mapping method [12] to solve non-linear advection problems.

The generalized advection equation is a hyperbolic conservation law, which models the transport phenomena of conserved quantities. The characteristic mapping method [12] has shown itself to be an excellent method for solving linear advection equations. It provides a better representation of solution with the help of the gradient information. It inherits most of its properties from its predecessor - the Gradient-Augmented Level Set method (GALS [13]).

The GALS was the first to use gradient information in the representation

of the solution to attain better accuracy in the solution. A similar attempt was made earlier by van Leer ([19], [20], [21] [22], [23]), but differently, as the gradients were stored and evolved as independent quantities. This was developed in the CIP method ([16], [17]). GALS provides a simpler scheme for allowing gradients to represent the sub-grid features, while keeping the stencils compact.

The characteristic mapping method [12] improves the sub-grid representation of the solution and removes the smoothness condition from the initial conditions. In this work we extend the characteristic mapping method to solve non-linear problems. We test the method on several test cases of 2D incompressible Euler equations and analyze results.

In Chapter 2, we introduce the linear advection problem and preceding work. A detailed description of the gradient-augmented level set method and characteristic mapping method is given in section 2.2 and 2.3 respectively. In Chapter 3 we discuss the key issues in extending the method to non-linear problems, and propose solutions to those issues. We include an analysis of a one-dimensional example test case. In Chapter 4 we apply the method for solving the 2D incompressible Euler equation for fluid flows and discuss the results obtained.

Chapter 2

Characteristic Mapping Method

2.1 Linear Advection Equation

Advection is the process of transport of quantities in a velocity field (fig. 2.1) [11]. The quantities could be properties of fluids such as mass or momentum or a general macroscopic quantity like traffic density. The velocity field could be a constant (fig. 2.1a), or a function of space and time (fig. 2.1b).

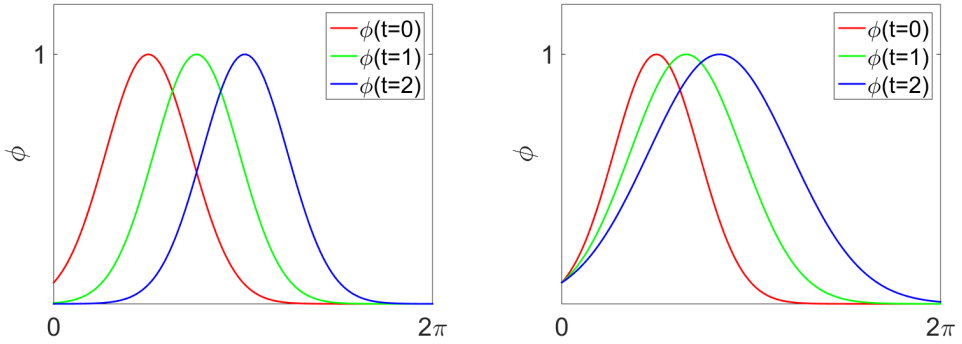
The following partial differential equation describes the phenomenon:

$$\phi_t(\vec{x},t) \ + \ (\vec{u}(\vec{x},t).\nabla)\ \phi(\vec{x},t) \ \ = \ \ 0 \tag{2.1}$$

for $\vec{x} \in \mathbb{R}^n$ and $t \in \mathbb{R}^+$, where $\phi : \mathbb{R}^n \times \mathbb{R}^+ \to \mathbb{R}$ is the scalar field representing a quantity and $\vec{u} : \mathbb{R}^n \times \mathbb{R}^+ \to \mathbb{R}^n$ is the advection velocity. In one dimension with constant velocity the equation simplifies to the following.

$$\phi_t \ + \ u\ \phi_x \ \ = \ \ 0 \tag{2.2}$$

The equation is called non-linear when the velocity field also depends on the scalar field ϕ, which makes the theory more involved as discussed later



(a) Advection in constant velocity field (b) Advection in variable velocity field

Figure 2.1: Advection of scalar quantity ϕ

in section 3.1. The equation can also be non-homogeneous, in which case it will no longer be the transportation of a conserved quantity.

The scope of this work spans the Cauchy problem of advection equation, later developing to fluid flow problems:

$$\left.\begin{aligned} \phi_t \ + \ (\vec{u}.\nabla)\phi \ &= \ 0 \\ \phi(\vec{x},0) \ &= \ g(\vec{x}) \end{aligned}\right\} \tag{2.3}$$

where $\vec{u} : \mathbb{R}^n \times \mathbb{R}^+ \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}$ are given functions representing the velocity field and initial conditions respectively. We are also given that g is continuously differentiable.

This problem is of central importance in the area of computational fluid dynamics, for it provides a minimal model for analysis of numerical methods aiming to solve complex fluid flow problems.

2.1.1 Method of Characteristics

The advection equation is hyperbolic in nature and has a finite propagation speed. The solution at each point travels along globally well defined curves

called characteristic curves. The methods of characteristics proposes, for the Cauchy problem for the non-homogenous advection problems of form :

$$\left.\begin{aligned} \phi_t \ + \ (\vec{u}.\nabla)\phi \ &= \ f \\ \phi(\vec{x},0) \ &= \ g(\vec{x}) \end{aligned}\right\} \tag{2.4}$$

the following solution:

$$\phi(\vec{\gamma}(s),t) \ = \ g(\vec{\gamma}(0)) \ + \ \int_0^t f(\vec{\gamma}(s),s)\ ds \tag{2.5}$$

$$\vec{\gamma}(s) \ = \ \vec{\gamma}(0) + \int_0^s \vec{u}(\vec{\gamma})\ ds \tag{2.6}$$

where $f : \mathbb{R}^n \times \mathbb{R}^+ \to \mathbb{R}$ is known as the source term.

The solution simplifies for the homogeneous case ($f(\vec{x},t) = 0$).

$$\phi(\vec{\gamma}(s),t) \ = \ g(\vec{\gamma}(0)) \tag{2.7}$$

$$\vec{\gamma}(s) \ = \ \vec{\gamma}(0) + \int_0^s \vec{u}(\vec{\gamma})\ ds \tag{2.8}$$

The solution at any point in space and time depends only on the point. More precisely, it remains constant in time along the locus $\vec{\gamma}(s)$ dictated by the following initial value problem.

$$\left.\begin{aligned} \frac{d\vec{\gamma}}{ds} \ &= \ \vec{u} \\ \vec{\gamma}(0) = \ \vec{x}_0 \end{aligned}\right\} \tag{2.9}$$

Fig. 2.2 shows the characteristic curves $\gamma(s)$, the solution to the initial value problem (equation. 2.9) for advection of scalar quantity ϕ with a velocity $u(x)$.

2.2 Gradient-Augmented Level-Set Method

The gradient-augmented level set method (GALS) [13] is a semi-Lagrangian approach for solving the Cauchy problem of advection equation using methods of characteristics.

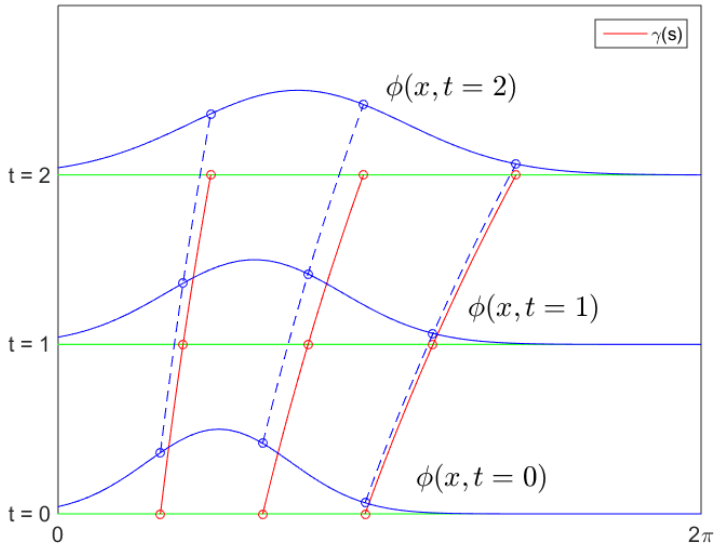


Figure 2.2: Representation of characteristic curves for one-dimensional advection problem

Given the Cauchy problem (equation (2.3)) on a discretized periodic domain with initial data and velocity function $\vec{u}(\vec{x}, t)$, the solution is represented by the values of ϕ on the grid points. GALS proposes to augment the function ϕ by its gradient $\vec{\psi} = \nabla\phi$. Knowledge of the values of the gradient along with the function provides a better representation of the solution using piece-wise Hermite interpolation, for it is a more accurate and continuously differentiable representation. Fig. (2.3) shows a comparison between the representation of curves using Hermite interpolation and linear interpolation. It can be noted that the former one is more accurate, smoother and contains features that the latter fails to capture.

The equation for the gradient $\vec{\psi}$ can be obtained from the advection

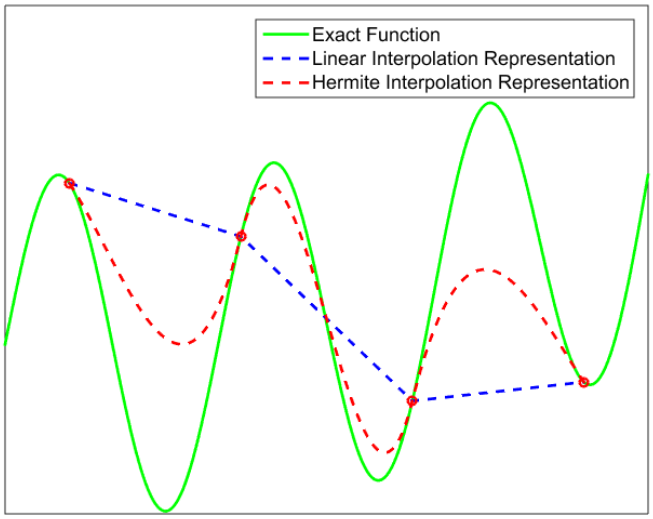


Figure 2.3: Comparison of piece-wise Hermite & linear interpolation

equation (3.1).

$$\begin{aligned} \nabla\phi_t \ + \ \nabla(\vec{u}.\nabla\phi) \ &= \ 0 \\ \vec{\psi}_t \ + \ \vec{u}.\nabla\vec{\psi} \ &= \ -\nabla\vec{u}\ \vec{\psi} \end{aligned} \tag{2.10}$$

Note that $\nabla\vec{u}$ is a gradient of velocity: a second-order tensor quantity. We will see later that we can solve the equation (2.10) without having to compute this quantity. The solution of the system now is the following :

$$\left.\begin{aligned} \frac{d\phi}{dt} \ &= \ 0 \\ \frac{d\psi}{dt} \ &= \ -\nabla\vec{u}\ \vec{\psi} \end{aligned}\right\} \ \text{along} \ \frac{d\vec{x}}{dt} \ = \ \vec{u} \tag{2.11}$$

2.2.1 Numerical Scheme

The problem is specified with a given initial condition $g(\vec{x})$ and a velocity field $\vec{u}(\vec{x},t)$, both in analytic form, and a set of points X_g forming a uniform grid

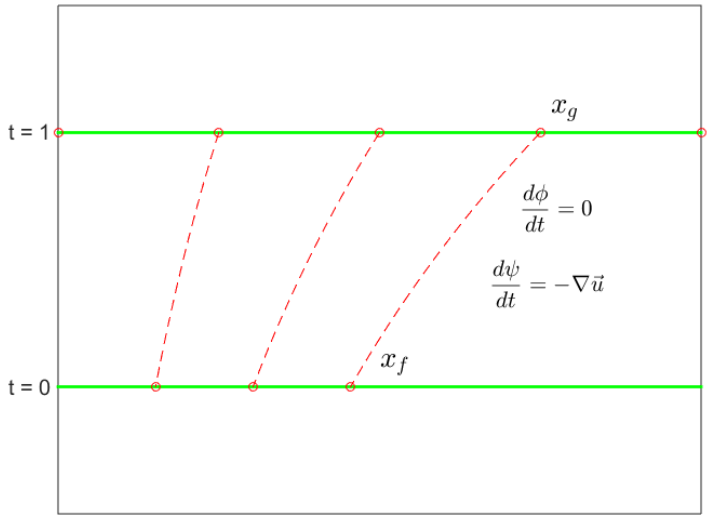


Figure 2.4: Representation of characteristic curves joining grid point x_g and foot point x_f

structure with grid interval h. Periodic boundary conditions are assumed. The initial time $t_0 = 0$ and a time step dt are asserted such that:

$$dt \leq \frac{0.5h}{\max\left(\|\vec{u}\|\right)} \tag{2.12}$$

For each iteration $\phi(:,t) \rightarrow \phi(:,t+dt)$, the following steps are performed.

GALS Iteration

1. Referring to fig. (2.4) for each grid point $\vec{x}_g \in X_g$, characteristic curves $\frac{d\vec{x}}{dt} = \vec{u}$ are numerically integrated backwards to obtain foot points x_f
2. $\phi(\vec{x}_f, t)$ is evaluated using Hermite interpolation from values $\phi(\vec{x}_g, t)$
3. $\phi(\vec{x}_g, t+dt) = \phi(\vec{x}_f, t)$
4. $\psi(\vec{x}_g, t+dt)$ is computed from the numerical integration of the following

equation along the characteristic curve:

$$\frac{d\psi}{dt} = -\nabla\vec{u}.\vec{\psi} \tag{2.13}$$

Hermite Interpolation

A piece-wise Hermite interpolation is used to interpolate $\phi(\vec{x}_f, t)$. Such an interpolation needs extra derivatives of ϕ, depending on the number of spatial dimensions of the problem. For problems in higher dimensions, the computational efficiency is improved using *epsilon derivatives* [3].

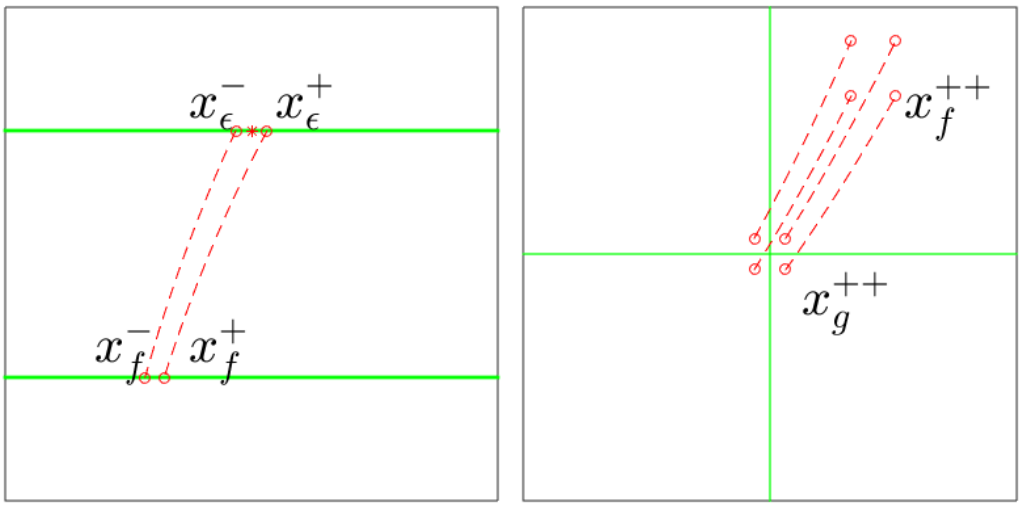
Epsilon Derivatives

Instead of calculating foot points for each grid point, a set of *epsilon points* (ϵ-points) $\vec{x}_\epsilon \in \{\vec{x}_g + \epsilon, \vec{x}_g - \epsilon\}^n$ for each grid point is taken, and their respective foot points are calculated, as shown in fig (2.5). Then the function values $\phi(\vec{x}_g, t + dt)$ and various derivatives can be approximated by a values of $\phi(\vec{x}_\epsilon)$. The following equation shows once such approximation for the one-dimensional case.

$$\begin{aligned}
\phi(x_g, t + dt) &= \frac{1}{2}\left(\phi(x_f^+, t) \; + \; \phi(x_f^-, t) \right) \\
\psi(x_g, t + dt) &= \frac{1}{2\epsilon}\left(\phi(x_f^+, t) \; - \; \phi(x_f^-, t) \right)
\end{aligned}$$

Temporal Integration

The numerical integration of the characteristic ODE is performed using the Shu-Osher temporal integration scheme [6], which is a strongly stability-preserving, globally third-order accurate scheme.



(a) ϵ-points in 1-dimension

(b) ϵ-points in 2-dimensions

Figure 2.5: Illustration of ϵ-points and characteristic curves passing through them

2.2.2 Accuracy & Stability

The spatial accuracy of the method is dictated by the interpolation scheme and the temporal accuracy is dictated naturally by the characteristic ODE integration scheme. With the choice of hermite interpolation and Shu-Osher numerical integration, the global accuracy of the method is third-order. This method is unconditionally stable. Also the method shows super-consistency under the approximations introduced due to the epsilon derivatives [13].

2.3 Characteristic Mapping Method

Since the solution of the Cauchy problem (equation (2.3)), the solution (equation (3.3)) can be written in the form of $\phi(\vec{x_g}, t) = g(\vec{x}_f)$ having a unique point $\vec{x}_f$ for each $\vec{x}_g$, the characteristic mapping method proposes to compute

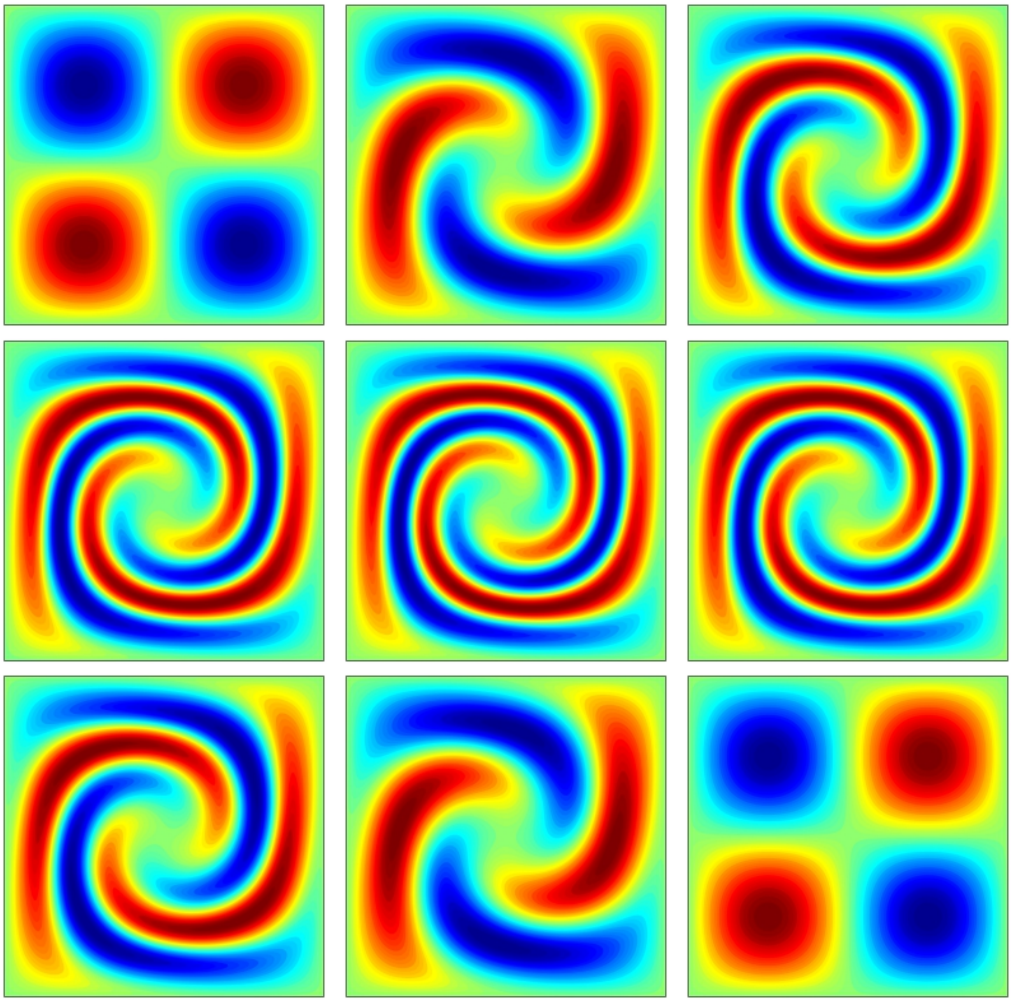


Figure 2.6: 2-D Advection using gradient augmented level set method

a map $\vec{\chi} : \mathbb{R}^n \times \mathbb{R}^+ \to \mathbb{R}^n$ such that :

$$\phi(\vec{x}, t) \;=\; g(\vec{\chi}(\vec{x}, t)) \tag{2.14}$$

From the global existence and uniqueness of a characteristic curve initial-value problem given in equation (2.9), $\vec{\chi}$ is guaranteed to be a bijective map $\vec{\chi}$.

The characteristic maps can be computed solving the equation:

$$\left.\begin{aligned} \vec{\chi}_t \ + \ (\vec{u}.\nabla)\vec{\chi} \ &= \ 0 \\ \vec{\chi}(\vec{x},0) \ &= \ \vec{x} \end{aligned}\right\} \tag{2.15}$$

The above Cauchy problem is solved using GALS. Transforming the problem into this form removes the smoothness condition from the initial condition g. This allows us to solve the advection problem with arbitrary initial conditions. Moreover, the analysis of the error of approximation shows a reduction for general case.

Fig. 2.7 shows the evolution of the characteristic map in the two-dimensional velocity field prescribed in equation 2.16.

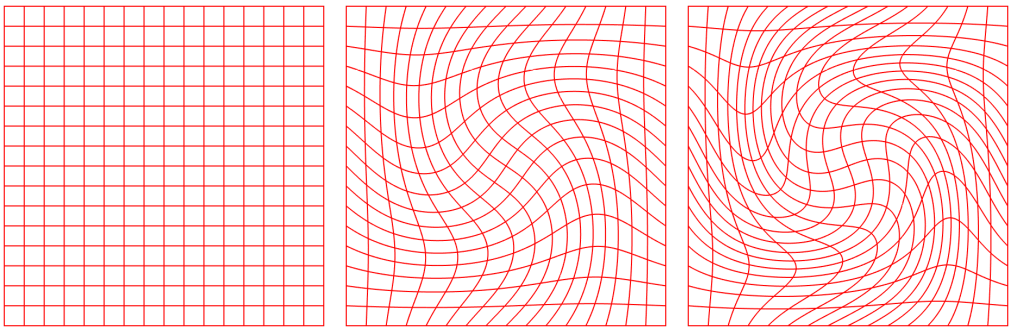
$$\left.\begin{aligned} u = \quad & \cos(\pi t)\sin^2(\pi x)\sin(2\pi y) \\ v = \quad & -\cos(\pi t)\sin^2(\pi y)\sin(2\pi x) \end{aligned}\right\} \tag{2.16}$$

2.3.1 Remapping

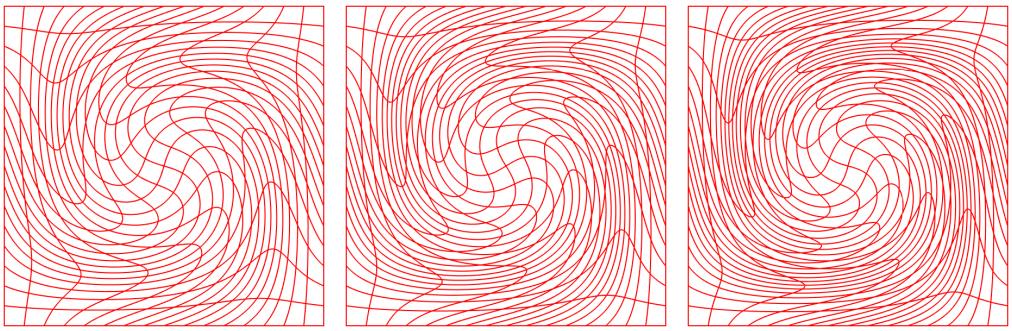
According to the definition of characteristic maps, they can be composed to form another map.

$$\vec{\chi} \ = \ \vec{\chi}_1 \circ \vec{\chi}_2 \tag{2.17}$$

The composition of two cubic maps gives a sixth order polynomial in each variable. Forming another cubic map from it is a projection, which can be computed on a finer grid, thus retaining more sub-grid features. This makes the method more robust because every time the global error of approximation grows, the map can be resampled on a finer map, and the problem can be reinitiated [12]. This allows us to represent the solution on a finer grid than the grid on which the computation of $\vec{\chi}$ is performed.

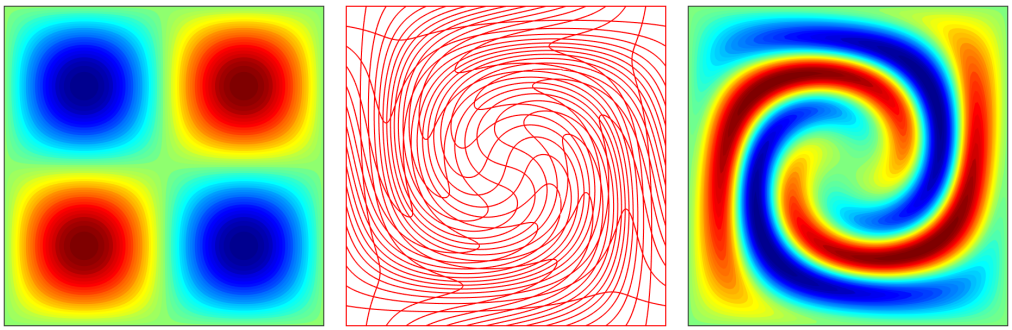


(a) $t = 0$ (b) $t = 0.1$ (c) $t = 0.2$



(d) $t = 0.3$ (e) $t = 0.4$ (f) $t = 0.5$

Figure 2.7: Advection of 2D map in velocity field in equation (2.16)



(a) $g(\vec{x})$ (b) $\vec{\chi}(\vec{x}, t)$ (c) $\phi(\vec{x}, t) = g(\vec{\chi}(\vec{x}, t))$

Figure 2.8: Applying map

Chapter 3

NonLinear Problems

3.1 Nonlinear Advection Equation

In non-linear advection equations the velocity field also depends on the advected scalar quantity ϕ. This can be thought of with the example of traffic density flow, where the speed of cars is governed by their density. The governing equation remains the same as the equation (3.1)

$$\phi_t(\vec{x},t) \ + \ (\vec{u}(\phi(\vec{x},t),\vec{x},t).\nabla)\ \phi(\vec{x},t) \ = \ 0 \tag{3.1}$$

with the velocity $\vec{u}(\vec{x},t,\phi(\vec{x},t))$ now being a function in higher dimensions.

$$\vec{u} : (\mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^+) \rightarrow \mathbb{R}^n$$

This change in the governing equation can lead to several physical phenomena like formation of compressive shock waves (like traffic jams in traffic flow models [7]) or expansive rarefaction waves, where the continuum assumptions are violated requiring intricate mathematical handling of the problem. But we limit our study to incompressible flows, a constraint which can be applied to higher than one dimensional flows, by imposing the following condition

$$\nabla.\vec{u} \ = \ 0 \tag{3.2}$$

The characteristic form of the solution of the equation has the same form as the linear case except that it is hard to solve and analyze and has no guarantee of global existence or uniqueness of characteristic curves.

$$\begin{aligned} \phi(\vec{\gamma}(s), t) &= g(\vec{\gamma}(0)) && (3.3) \\ \vec{\gamma}(s) &= \vec{\gamma}(0) + \int_0^s \vec{u}(\phi(\vec{\gamma}(s), s), \vec{\gamma}(s), s)\ ds && (3.4) \end{aligned}$$

3.2 Burgers' Equation

Burgers' equation is a one dimensional scalar conservation equation, which provides a canonical case for understanding the fundamental complications arising in the CM method

The one-dimensional scalar conservation law has the following form :

$$\phi_t + f_x(\phi) = 0 \quad (3.5)$$

where f is the flow flux of the flowing quantity ϕ. Applying the chain-rule we get a non-linear advection problem :

$$\phi_t + f_\phi \phi_x = 0$$

with advection velocity being $u = f_\phi$.

Choosing the flux as the energy estimate of the quantity $f = \frac{1}{2}\phi^2$, we get the burgers' equation:

$$\phi_t + \phi\phi_x = 0 \quad (3.6)$$

The method of characteristics applies directly to the computation of the solutions until a feature such as shocks is formed. Fig. (3.1) shows the example of a shock formation in the solution of the Burgers' equation with given

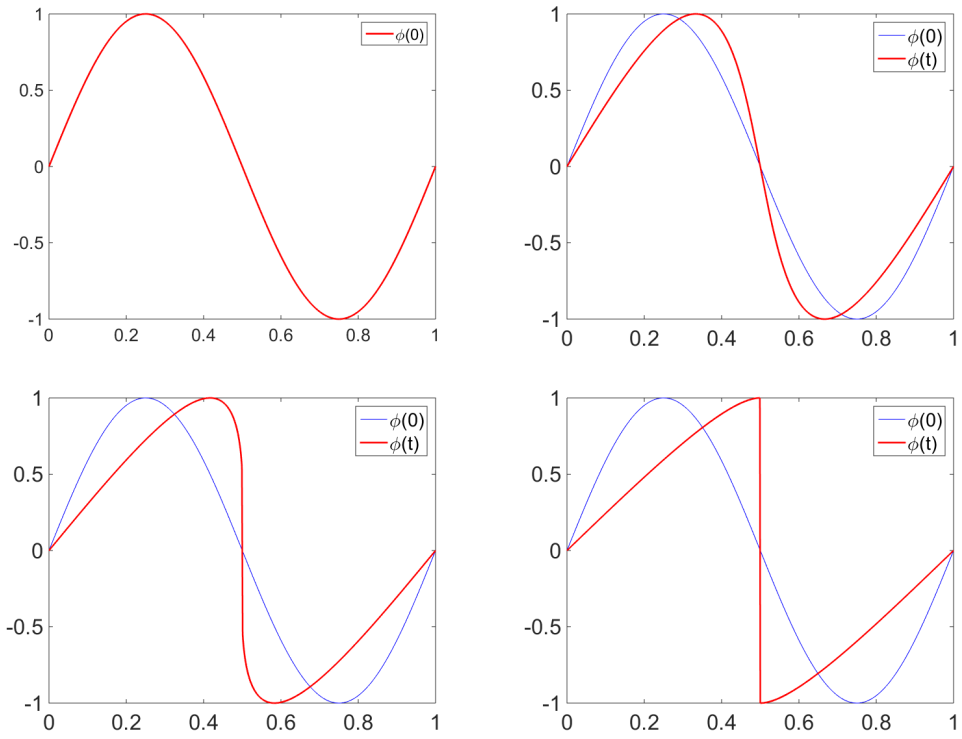


Figure 3.1: Solution of Burgers' equation with initial data $g(x) = \sin(x) \quad \forall \quad x \in [0, 2\pi]$ and periodic boundary conditions forming a shock

initial data $g(x) = \sin(x) \quad \forall \quad x \in [0, 2\pi]$ and periodic boundary conditions. Once the shock is formed, a more involved theory comes and allows us to numerically regularize the problem.

To avoid shock formation, we take the following variation of the nonlinear advection equation.

$$\phi_t \ + \ (\alpha.\phi.\cos(t))\phi_x \ = \ 0 \tag{3.7}$$

with a parameter α governing the shock formation. A value of α can be chosen such that a shock-like feature producing an artefact of discontinuity will be formed and retraced back, as shown in fig. (3.2). The motivation

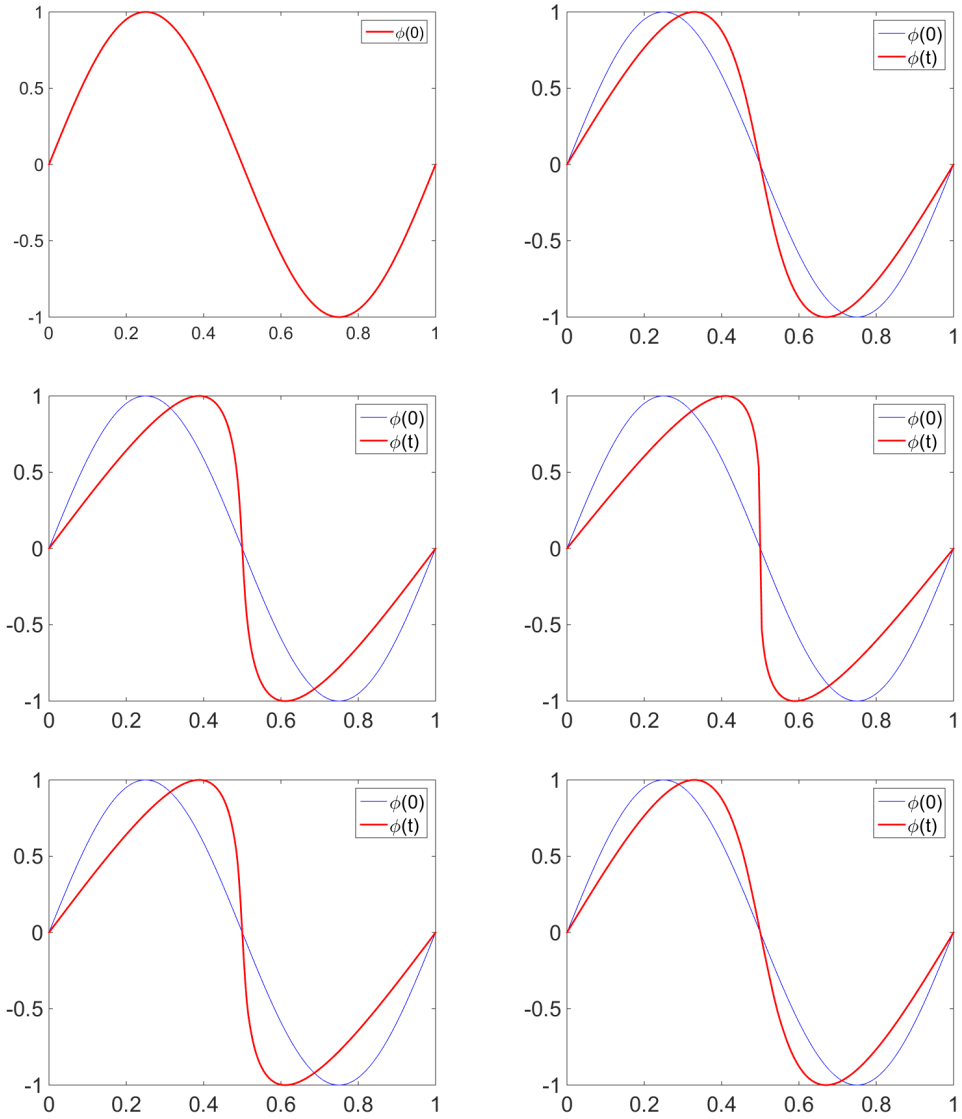


Figure 3.2: Solution of equation (3.7) with $\alpha = 0.16$, initial data $g(x) = \sin(x) \quad \forall \quad x \in [0, 2\pi]$ and periodic boundary conditions

for choosing the example is to have a case without discontinuities in which the method of characteristics can be applied, while still having stiff regions, which are likely to form in real fluid flow problems in higher dimensions, even with the incompressibility constraint.

It is important to note that a shock differs from a stiff structure in that the former has characteristic curves colliding with each other, while the latter has characteristic curves coming closer to each other.

3.3 Characteristic Mapping Method for Nonlinear Problems

The foremost issue in solving the nonlinear advection problems using the characteristic mapping method is that the velocity function is not know everywhere to solve for characteristic ODE from equation (2.9).

As described in section 2.2.1, for advecting the map numerically $\chi(\vec{x}, t_n) \rightarrow \chi(\vec{x}, t_{n+1})$ the backwards temporal integration for calculating the foot point for each grid point requires a known velocity field for all $\vec{x}$ and $t \in \{t_n, t_{n+1}\}$. But in the nonlinear advection case, since the velocity is a function of the advecting field, it is known only from the data values at grid point of previous times.

This requires introducing another approach of temporal integration listed in algorithms 1 and 2.

Algorithm 1 One-Stage Fixed Point Iteration Backward Integration

1: $\vec{x}_f = \vec{x}_g$
2: **while** $|\vec{x}_f + dt.\vec{u}(\vec{x}_f, t) - \vec{x}_g| \geq \epsilon$ **do**
3: $\quad \vec{x}_f = \vec{x}_g - dt.\vec{u}(\vec{x}_f, t)$
4: **end while**

Here, the velocity $\vec{u}(\vec{x}, t)$ is shorthand for $\vec{u}(\phi(\vec{x}, t), \vec{x}, t)$ and it can be computed from the interpolation of $\phi(\vec{x}_g, t)$ from $\phi(\vec{x}_g, t-dt)$ which is known.

Algorithm 2 Two-Stage Fixed Point Iteration Backward Integration

1: $\vec{x}_f = \vec{x}_g$

2: $\vec{x}_{ff} = \vec{x}_g$

3: **while** $|\vec{x}_f + dt\left(\frac{3}{2}\vec{u}(\vec{x}_f, t) - \frac{1}{2}\vec{u}(\vec{x}_{ff}, t - dt)\right) - \vec{x}_g| \geq \epsilon$ **do**

4: $\quad \vec{x}_{ff} = \vec{x}_f - dt.\vec{u}(\vec{x}_{ff}, t - dt)$

5: $\quad \vec{x}_f = \vec{x}_g - dt\left(\frac{3}{2}\vec{u}(\vec{x}_f, t) - \frac{1}{2}\vec{u}(\vec{x}_{ff}, t - dt)\right)$

6: **end while**

3.3.1 Overshooting

Hermite interpolation doesn't preserve monotonicity ([4], [8] [9] [2]). Thus, it can lead to overshooting for stiff data (fig. 3.3). This becomes a serious while solving nonlinear problems using the characteristic mapping method.

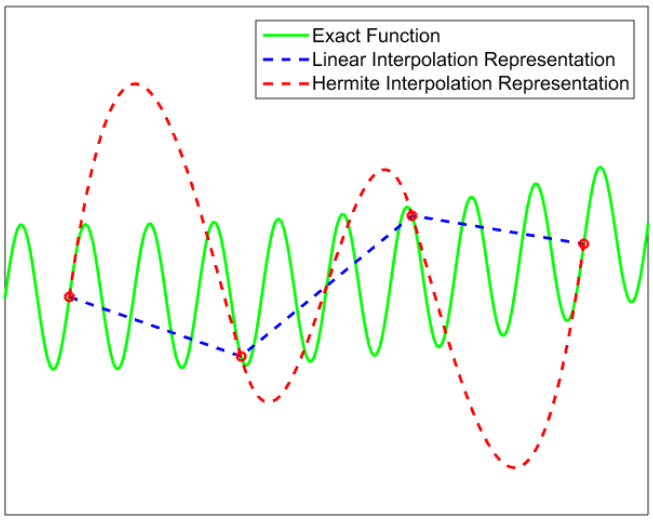


Figure 3.3: Overshooting in hermite interpolation

The problem can also arise in linear problems but need more reflection for non-linear problem for the following reasons.

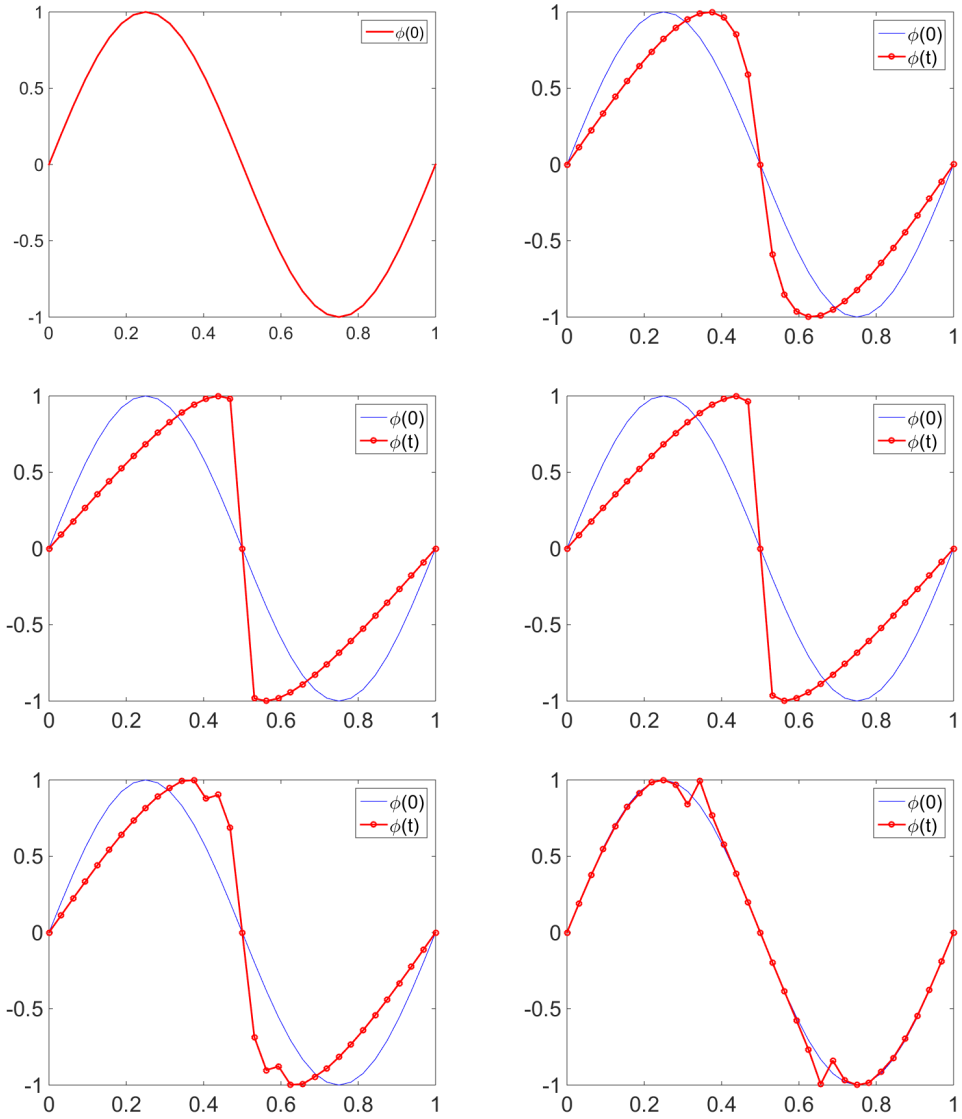


Figure 3.4: Numerical Solution of equation (3.7) with $\alpha = 0.2$, initial data $g(x) = \sin(x) \quad \forall \quad x \in [0, 2\pi]$ and periodic boundary conditions

- In non-linear problems, the velocity is the function of the solution ϕ itself. An overshoot in the solution adds oscillations to the velocity field causing more overshooting in ϕ as shown in fig. (3.4)

- A stiffness of solution for linear problems can be estimated beforehand

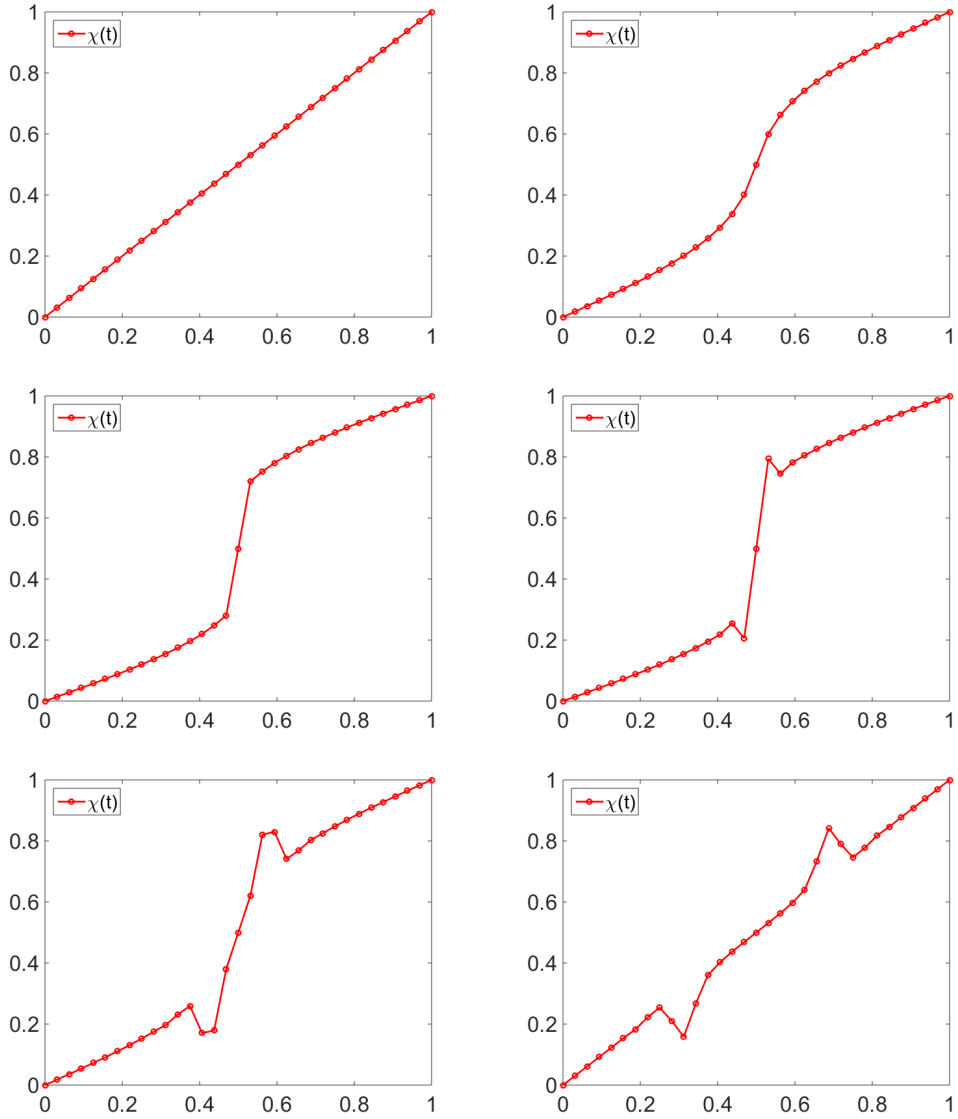


Figure 3.5: Numerical Solution χ of equation (3.7) with $\alpha = 0.2$

most of the time, thus allowing a finer grid to represent the solution.

- Stiffness of solution comes naturally for non-linear problems. Incompressibility conditions avoid shocks, but the stiffness is likely to arise depending on initial data. For example, in Burgers' equation any negative gradient in the initial data would form a shock later in time.

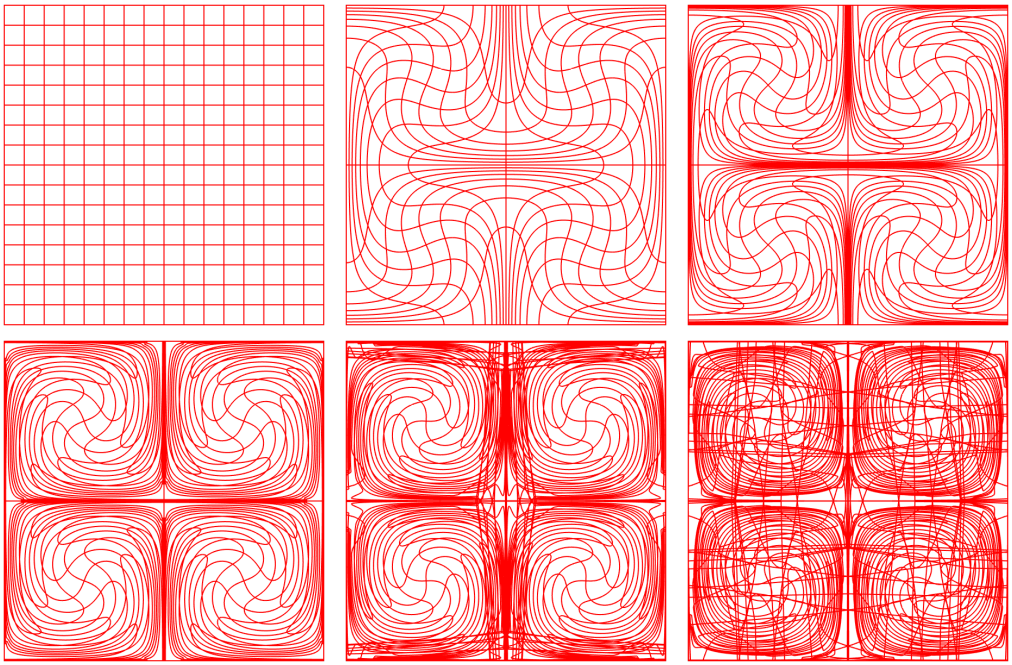


Figure 3.6: Overshooting in map $\vec{\chi}$ in 2-dimensions

The example in equation (3.7) demonstrates a minimal case with a control on the stiffness of the problem with the parameter α. When the solution becomes stiff in a region, if the grid is not fine enough, the solution shows an overshoot. Figure (3.4) shows a case with $\alpha = 0.2$, grid size $N = 32$, initial data $g(x) = \sin(x) \ \ \forall \ \ x \in [0, 2\pi]$ and periodic boundary conditions.

The propagation of oscillations in the map $\vec{\chi}$ is shown in fig. (3.5). It is important to note here that a characteristic map by definition needs to be a one-to-one map, which in this case is violated, suggesting that the solution is unreliable.

In two-dimensions, a map at any given time $\vec{\chi}$ could have even more complicated overshooting features. Fig. (3.6) shows a map evolved in velocity

field :

$$\vec{u} \;=\; \begin{bmatrix} -\sin(x_1)\cos(x_2) \\ \\ \cos(x_1)\sin(x_2) \end{bmatrix} \tag{3.8}$$

for $\vec{x} = \{x_1, x_2\} \in [0, 2\pi] \times [0, 2\pi]$. This problem is linear but motivated from the non-linear problem of Taylor-Green vortex flow.

3.3.2 Solution to Overshooting : Map-Stacking

Preserving the incompressibilty of the flow is equivalent to imposing the following constraint on the map:

$$\det|\nabla\vec{\chi}| \;=\; 1 \tag{3.9}$$

where $\nabla\vec{\chi}$, gradient of the map, is a second-order tensor quantity, and also in this case the Jacobian matrix $\dfrac{\partial\chi_i}{\partial x_j}$.

The definition of map $\vec{\chi} : \mathbb{R}^n \to \mathbb{R}^n$ is a column of n cubic hermite interpolants independent to each other. Such a definition is not capable of applying to the map itself. The incompressibility of the flow comes from the incompressibility constraint in the velocity field (equation (3.2)), But incompressibility in the solution $\vec{\chi}$ is the accumulation of numerical error, eventually violating equation (3.9) significantly. The problem becomes even more serious when $\det|\nabla\vec{\chi}| \to 0$, and the map is no longer one-to-one.

To avoid this, a stronger form of remapping is suggested. Whenever the error $e \;=\; \left|\left(\det\left|\vec{\nabla\chi}\right| \;-\; 1\right)\right|$ is more than a certain threshold τ, the map is pushed to a stack of maps that is stored and the problem is re-initiated with the identity map $\vec{\chi}(\vec{x}, t) = \vec{x}$. The solution can be represented by a composition of all the maps in the stack. The steps are shown in algorithm (3).

Algorithm 3 Map Stacking Scheme

1: $n,\ N,\ \lambda$
2: $j = 0$
3: **while** $n < N$ **do**
4: advect $\vec{\chi}(:, t_n) \rightarrow \vec{\chi}(:, t_{n+1})$
5: **if** $\left|\left(\left|\vec{\nabla}\chi\right| - 1\right)\right| > \lambda$ **then**
6: $\vec{\chi}_j = \vec{\chi}(:, t_{n+1})$
7: $j = j + 1$
8: $\vec{\chi}(:, t_{n+1}) = \vec{\mathcal{I}}$
9: **end if**
10: **end while**
11: $\phi = \vec{\chi} \circ \vec{\chi}_{j-1} \circ \vec{\chi}_{j-2} \circ ... \circ \vec{\chi}_0 \circ g$

This differs conceptually from the fine grid remapping because a remapping step: $\vec{\chi}^f = \vec{\chi}^f \circ \vec{\chi}_j$ projects the composition of two cubic maps forming a ninth-order representation to a cubic map, thus loosing some precision. While, the map-stacking builds the order-of-representation by a degree of three with each new map in the stack without loosing anything in the projection. An example of representation of stiff features using map-stacking method is shown in fig. 3.7. This solves the problem at the cost of computing composition of map whenever the solution is required. Figure 3.7 shown an the solution of equation (3.7) using map-stacking.

The method can be generalized to higher dimensions as well. An example in 2-dimensions of a map advection in velocity field given in equation (3.8) is shown in fig. 3.8 which is a improvement in the solution shown in fig. 3.6.

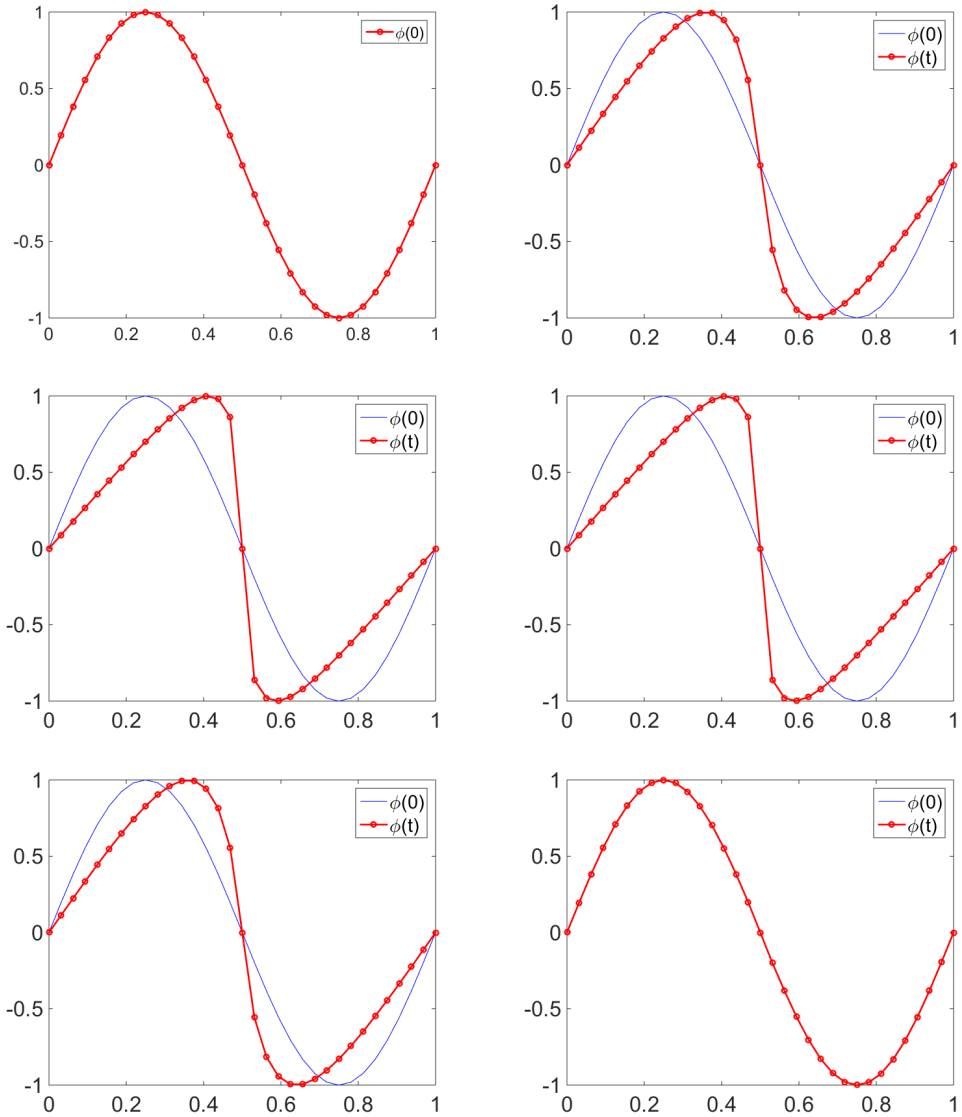


Figure 3.7: Numerical Solution χ of equation (3.7) using map-stacking technique with $\alpha = 0.2$, initial data $g(x) = \sin(x) \quad \forall \quad x \in [0, 2\pi]$ and periodic boundary conditions

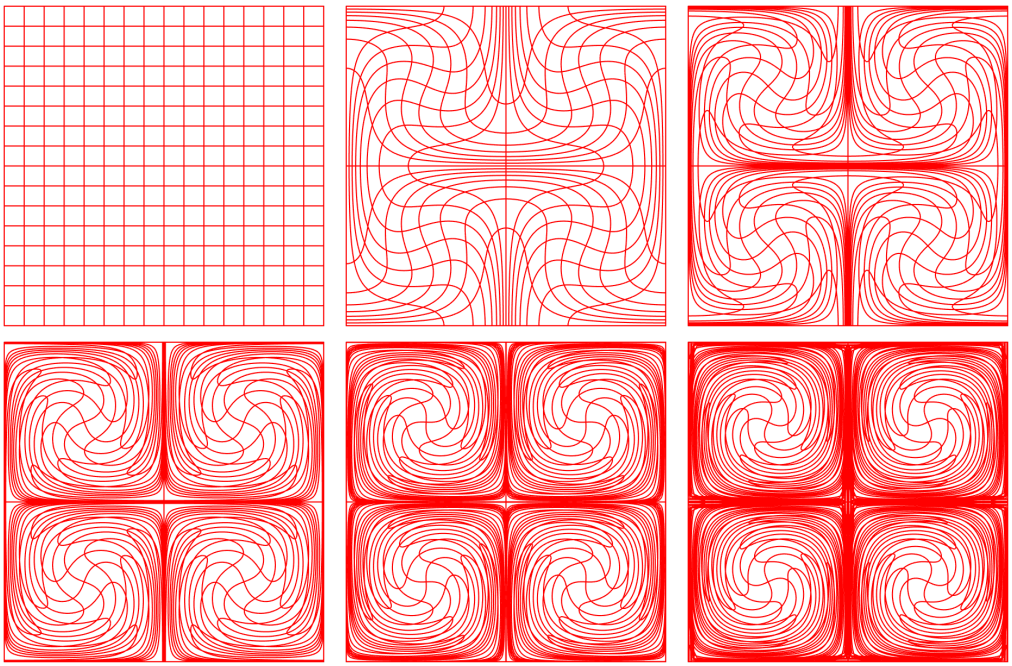


Figure 3.8: Map-stacking in two-dimensions

Chapter 4

2D Euler Equation

4.1 2D Incompressible Euler Equation

Euler equations are a set of non-linear hyperbolic conservation laws governing the flow of adiabatic and inviscid flow. Along with the incompressibility constraint (equation (3.2)), it is a very reasonable model for low Mach number flows. Euler equation provides an excellent case for testing characteristic mapping method on nonlinear fluid flow problems Applying the incompressibility constraints, the equation simplifies to the following:

$$\left.\begin{aligned} \vec{u}_t \ + \ (\vec{u}.\nabla)\ \vec{u} \ &= \ -\frac{1}{\rho}\nabla p \\ \nabla.\vec{u} = \ &0 \end{aligned}\right\} \tag{4.1}$$

where $\vec{u}$ is the velocity field, ρ is the density, and p is pressure. The incompressibility condition implies that the volume spanned by a certain fluid element remains constant in time.

Introducing vorticity $\vec{\omega} = \nabla \times \vec{u}$, the equation can be written in form of a non-homogenous nonlinear transport equation called vorticity transport

equation obtained by taking the curl of the equation (4.3).

$$\vec{\omega}_t \ + \ (\vec{u}.\nabla)\ \vec{\omega} \ \ = \ \ (\vec{\omega}.\nabla)\vec{u} \ - \ \vec{\omega}(\nabla.\vec{u}) + \frac{1}{\rho^2}\nabla p \times \nabla\rho \tag{4.2}$$

Further assuming the flow to be 2-dimensional and barotropic ($\nabla p \times \nabla \rho = 0$), the equation simplifies to homogeneous transport equation.

$$\left.\begin{aligned} \vec{\omega}_t \ + \ (\vec{u}.\nabla)\ \vec{\omega} = \ \ 0 \\ \nabla.\vec{u} = \ \ 0 \end{aligned}\right\} \tag{4.3}$$

The equation is solved together with the incompressibility constraint along with initial data of $\vec{u}$ given and in periodic boundary conditions.

4.2 Characteristic Mapping Method for Euler Equation

While solving the equation (**??**), we need a relation to compute advecting velocity field such that the incompressibility constraint is satisfied. The following construction allows that.

$$\left.\begin{aligned} \varphi \ \ = \ \ \Delta^{-1}\vec{\omega} \\ \vec{u} \ \ = \ \ \nabla^{\perp}\varphi \end{aligned}\right\} \tag{4.4}$$

where φ is an intermediate stream function and $\nabla^{\perp}$ stands for perpendicular gradient $(\frac{\partial}{\partial y}, -\frac{\partial}{\partial x})$.

Algorithm (4) shows the basic overview of the steps of the characteristic mapping method. The problem is initialized with an initial Cauchy data ω_0 defined on a set of points $X_g = \{\vec{x}_g\}$ forming a uniform grid with grid spacing h and periodic boundary conditions. A characteristic map $\vec{\chi} : \mathbb{T}^2 \to \mathbb{T}^2$, set to identity $\vec{\chi}(\vec{x}, 0) = \vec{x}$ is taken as the initial condition for the map. A

time step dt is chosen such that constraint in equation (2.12) is followed throughout the desired running time of the simulation and time t_n is defined to be $n.dt$

Algorithm 4 Characteristic mapping method for solving incompressible Euler equation

1: $\vec{\chi}(\vec{x}, 0) = \vec{x}$
2: ω_0
3: $n = 0, t_0 = 0$
4: N, dt
5: **while** $n < N$ **do**
6: $\quad \omega_n(\vec{x}) = \omega_0(\vec{\chi_n}(\vec{x}, t_n))$
7: $\quad$ advect $\vec{\chi}(:, t_n) \rightarrow \vec{\chi}(:, t_{n+1})$ using ω_n
8: $\quad n = n + 1$
9: **end while**
10: $\omega_N(\vec{x}) = \omega_0(\vec{\chi}(\vec{x}, t_N))$

The advection of $\vec{\chi}(:, t_n) \rightarrow \vec{\chi}(:, t_{n+1})$ is the detailed step of the algorithm and requires a mix of few steps discussed so far. Most importantly, advection of the map needs a velocity field defined on the grid X_g. This is attained by forming a Hermite interpolant on the grid using the construction shown in equation (4.4). Firstly, the vorticity at any time step $\omega_n(\vec{x})$ defined on the grid is transformed to a Fourier representation $\hat{\omega}_n(\vec{\zeta})$ in the frequency space using a fast Fourier transform. This is for the faster and more accurate calculation of the stream function and its derivatives. The set of values of stream function and its derivatives on the grid $S = \{(\varphi(\vec{x}),\ \partial_x\varphi(\vec{x}),\ \partial_y\varphi(\vec{x}),\ \partial_x\partial_y\varphi(\vec{x}))\ \ \forall\vec{x} \in X_g\}$ forms a Hermite interpolant $\mathcal{H}_S$ from which the velocity can be optained. The steps in the process are detailed in the algorithm (5) given below.

Algorithm 5 Computation of velocity from vorticity data

1: $\hat{\omega}_n(\vec{\zeta}) \;=\; \mathcal{F}(\omega_n(\vec{x}))$

2: $\hat{\varphi}_n(\vec{\zeta}) \;=\; (\mathbf{i}\vec{\zeta})^{-2}\hat{\omega}_n(\vec{\zeta}) \;=\; -(\vec{\zeta}.\vec{\zeta})\hat{\omega}_n(\vec{\zeta})$

3: $\partial_x\hat{\varphi}_n(\vec{\zeta}) \;=\; (\mathbf{i}\zeta_x)\hat{\omega}_n(\vec{\zeta})$

4: $\partial_y\hat{\varphi}_n(\vec{\zeta}) \;=\; (\mathbf{i}\zeta_y)\hat{\omega}_n(\vec{\zeta})$

5: $\partial_x\partial_y\hat{\varphi}_n(\vec{\zeta}) \;=\; -(\zeta_x\zeta_y)\hat{\omega}_n(\vec{\zeta})$

6: $\varphi_n(\vec{x}) \;=\; \mathcal{F}^{-1}(\hat{\varphi}_n(\vec{\zeta}))$

7: $\partial_x\varphi_n(\vec{x}) \;=\; \mathcal{F}^{-1}(\partial_x\hat{\varphi}_n(\vec{\zeta}))$

8: $\partial_y\varphi_n(\vec{x}) \;=\; \mathcal{F}^{-1}(\partial_y\hat{\varphi}_n(\vec{\zeta}))$

9: $\partial_x\partial_y\varphi_n(\vec{x}) \;=\; \mathcal{F}^{-1}(\partial_x\partial_x\hat{\varphi}_n(\vec{\zeta}))$

10: $u(\vec{x}) \;=\; \partial_y\mathcal{H}_S$

11: $v(\vec{x}) \;=\; -\partial_x\mathcal{H}_S$

Here, $\mathcal{F}$ and $\mathcal{F}^{-1}$ represents the Fourier and the inverse Fourier transform respectively, and $\mathbf{i}$ stands for the imaginary root. Steps (2) to (5) shows the fourier differentiation operations assuming the function $\omega_n(\vec{x})$ is defined on the domain $[0, 2\pi]^2$. These operations will need a scaling factor in case of other rectangular domains. The algorithm shows that the velocity can be computed by differentiating the Hermite interpolant. This process allows to compute velocity $\vec{u}(\vec{x}) = [u(\vec{x}), v(\vec{x})]$ at any point in the spanned domain while appling the incompressibility constraint to third order accuracy.

Once the velocity field is computed, algorithm (1) or (2) can be implemented to advect the map. Note that the map itself is a Hermite representation and defined by by the set of values: $\{(\vec{\chi}(\vec{x}),\ \partial_x\vec{\chi}(\vec{x}),\ \partial_y\vec{\chi}(\vec{x}),\ \partial_x\partial_y\vec{\chi}(\vec{x}))\}$ on the grid points. Each advection step updates the values following the steps shown int the algorithm (6).

A small value of ϵ guarantees the order of the error of approximations

Algorithm 6 Advection Map $\vec{\chi}(:,t_n) \to \vec{\chi}(:,t_{n+1})$

1: $\epsilon = 10^{-6}$

2: $\{\epsilon_x^0,\ \epsilon_x^1,\ \epsilon_x^2,\ \epsilon_x^3\} = \{\epsilon,\ \epsilon,\ -\epsilon,\ -\epsilon\}$

3: $\{\epsilon_y^0,\ \epsilon_y^1,\ \epsilon_y^2,\ \epsilon_y^3\} = \{\epsilon,\ -\epsilon,\ \epsilon,\ -\epsilon\}$

4: **for** each $\vec{x}_g = (x, y) \in X_g$ **do**

5: **for** each $k \in [0, 1, 2, 3]$ **do**

6: $\vec{x}_\epsilon^k = (x + \epsilon_x^k, y + \epsilon_y^k)$

7: Calculate foot point : $\vec{x}_f^k$

8: **end for**

9: Update :

$$\left.\begin{aligned}
\vec{\chi}(\vec{x}_g, t_{n+1}) &= \frac{1}{4}\left(\vec{\chi}(\vec{x}_f^0, t_n) + \vec{\chi}(\vec{x}_f^1, t_n) + \vec{\chi}(\vec{x}_f^2, t_n) + \vec{\chi}(\vec{x}_f^3, t_n)\right) \\
\partial_x\vec{\chi}(\vec{x}_g, t_{n+1}) &= \frac{1}{4\epsilon}\left(\vec{\chi}(\vec{x}_f^0, t_n) + \vec{\chi}(\vec{x}_f^1, t_n) - \vec{\chi}(\vec{x}_f^2, t_n) - \vec{\chi}(\vec{x}_f^3, t_n)\right) \\
\partial_y\vec{\chi}(\vec{x}_g, t_{n+1}) &= \frac{1}{4\epsilon}\left(\vec{\chi}(\vec{x}_f^0, t_n) - \vec{\chi}(\vec{x}_f^1, t_n) + \vec{\chi}(\vec{x}_f^2, t_n) - \vec{\chi}(\vec{x}_f^3, t_n)\right) \\
\partial_x\partial_y\vec{\chi}(\vec{x}_g, t_{n+1}) &= \frac{1}{4\epsilon^2}\left(\vec{\chi}(\vec{x}_f^0, t_n) - \vec{\chi}(\vec{x}_f^1, t_n) - \vec{\chi}(\vec{x}_f^2, t_n) + \vec{\chi}(\vec{x}_f^3, t_n)\right)
\end{aligned}\right\} \quad (4.5)$$

10: **end for**

in equation (4.5) to be smaller than the overall spacial order of the method itself [13].

These unit operations are performed for each iteration, along with map-stacking technique. As the stack of map grows larger, it becomes computationally costly to recompute ω_n at each time step. To avoid this, the initial condition is stored on a very fine grid and updated every time a map is added to the stack. This allows velocity field to be computed at each time step without having to compose the maps in the stack. Although, the final solution whenever needed to be computed make use of the analytical form of initial data. Note that this algorithm achieves exponential resolution (each remapping is a composition with a cubic) in linear time.

4.3 Results & Discussion

In this section we discuss the implementation of characteristic mapping method for solving 2-dimensional incompressible Euler equations for two cases: vortex merging and shear layer. We also present the analysis of properties of the method using results from the vortex merging phenomenon and homogeneous-isotropic flow.

The implementation of the method is done in *C++* language with *CUDA* platform. All the following simulations are run on a *Dell-Inspiron 7000* laptop with the *Intel i7-4510U* CPU and the *Nvidia 750M GPU*.

4.3.1 Vortex Merging

We initialize the simulation with the analytic vorticity field generated by superimposing two Gaussian functions centered at two different points (eq. 4.6) as shown in the figure (4.1). We maintain the periodicity at the boundaries

by using the padding technique described in algorithm (7).

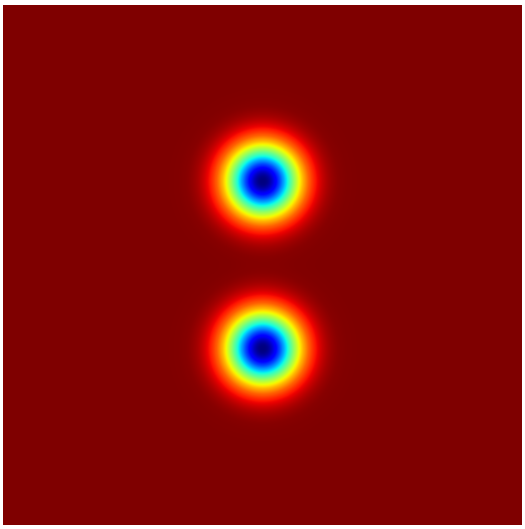


Figure 4.1: Initial condition for vortex merging

$$\begin{aligned} f(x,y) &= \sin^2(x)\sin^2(y)\left(\exp\left(5\left((x-\pi)^2+(y-\frac{\pi}{3})^2\right)\right)\right. \\ &+ \left.\exp\left(5\left((x-\pi)^2+(y-\frac{2\pi}{3})^2\right)\right)\right) \\ &\forall\ (x,y)\in[0,2\pi]\times[0,2\pi] \end{aligned} \tag{4.6}$$

Algorithm 7 Padding of Initial Conditions for Periodicity at boundaries

1: $\omega_0(x,t) = 0$
2: **for** $i = \{-1,0,1\}$ **do**
3: **for** $i = \{-1,0,1\}$ **do**
4: $\omega_0(x,t) = \omega_0(x,t) + f(x+i2\pi, y+j2\pi)$
5: **end for**
6: **end for**

We run the simulation on a computational grid of size 128×128 with a time step $dt = 0.01$ seconds. The two vortex rotates, shear and merge to

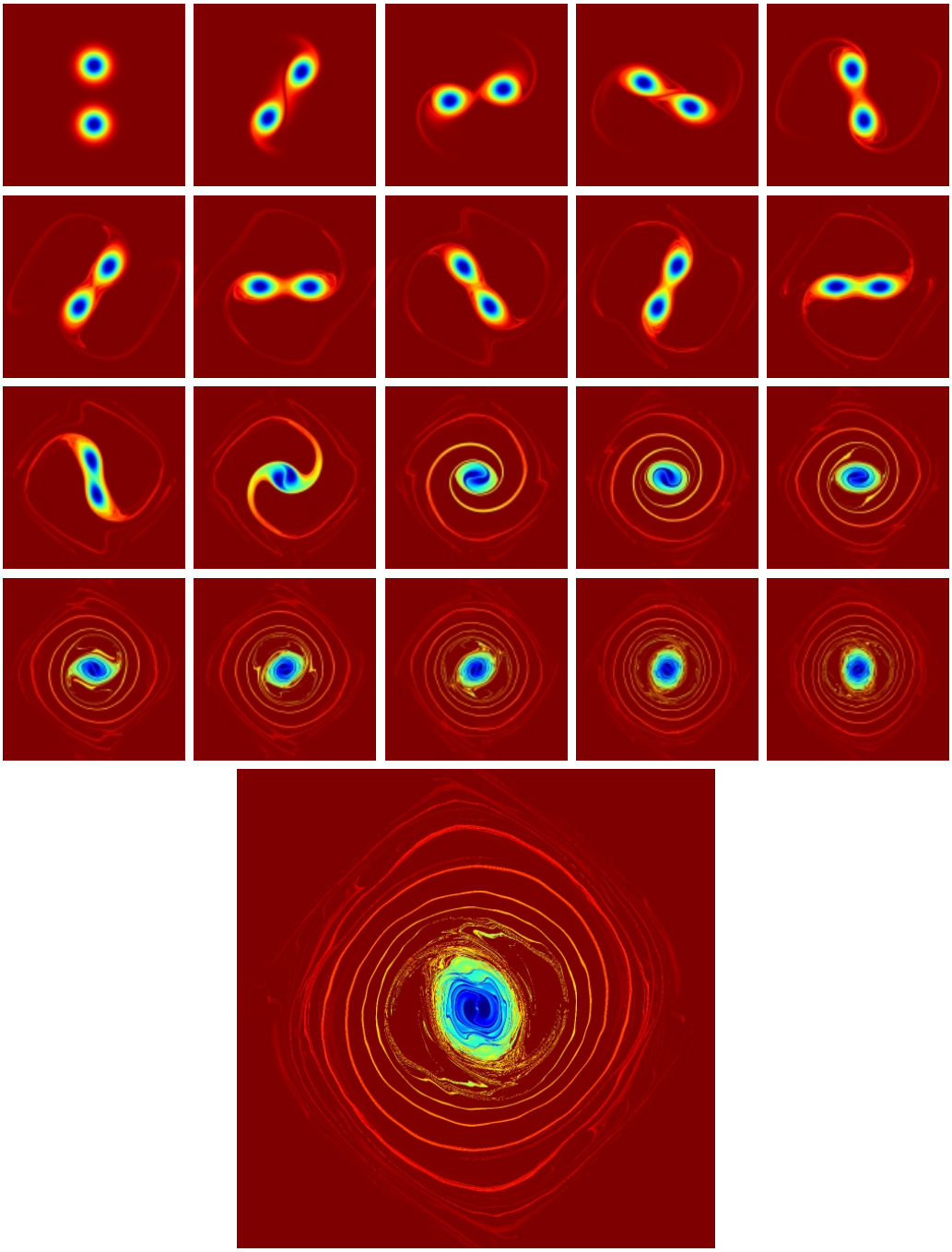


Figure 4.2: Vortex merging (each image is produced after 2500 iterations of 0.01 seconds each)

form a larger rotating vortex. This is shown in the figure (4.2).

In the next sections, we make a few remarks about the properties of the method.

Smallest Length Scales

The unique property of the characteristic mapping method is its ability to represent small length scales in the solution, which are smaller than the resolution of the computational grid. This comes from the fact the solution is represented by a stack of characteristic maps which are to be composed with the analytical initial condition. Using this, the final solution can be obtained by sampling on arbitrarily fine grid.

In this section we sample a small region in the domain for different length scales. This allows for zooming in the solution as shown in figure (4.4). We notice that the smallest available feature in the solution is seen at a length scale of order 10^{-12}. as shown in the figure (4.3).

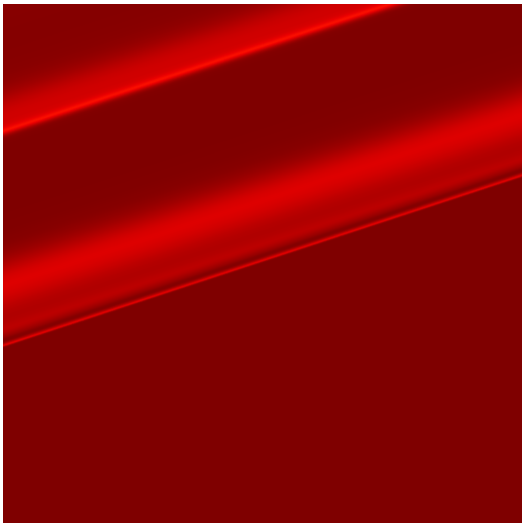
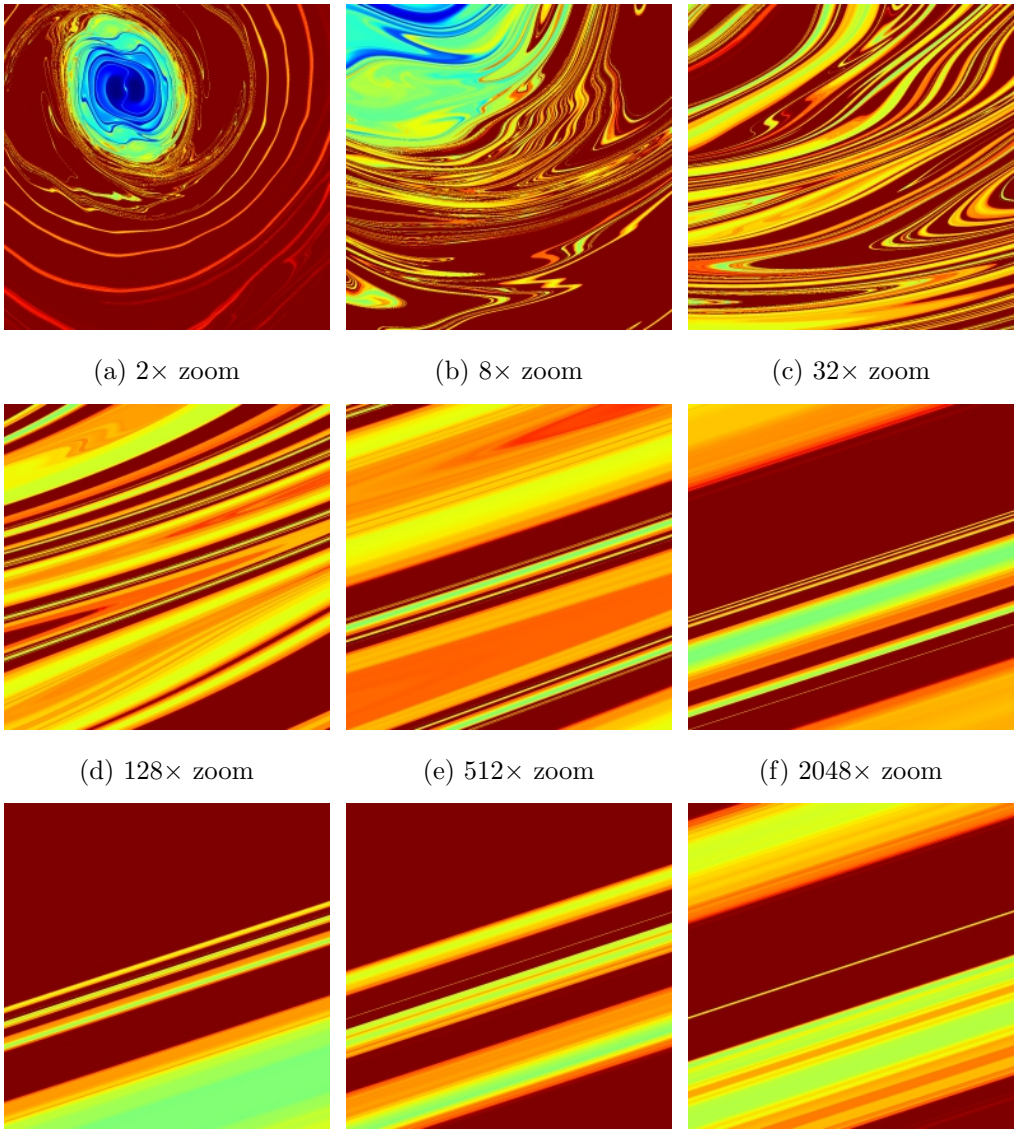


Figure 4.3: Solution sampled at length scales of order 10^{-12}



(a) 2× zoom (b) 8× zoom (c) 32× zoom

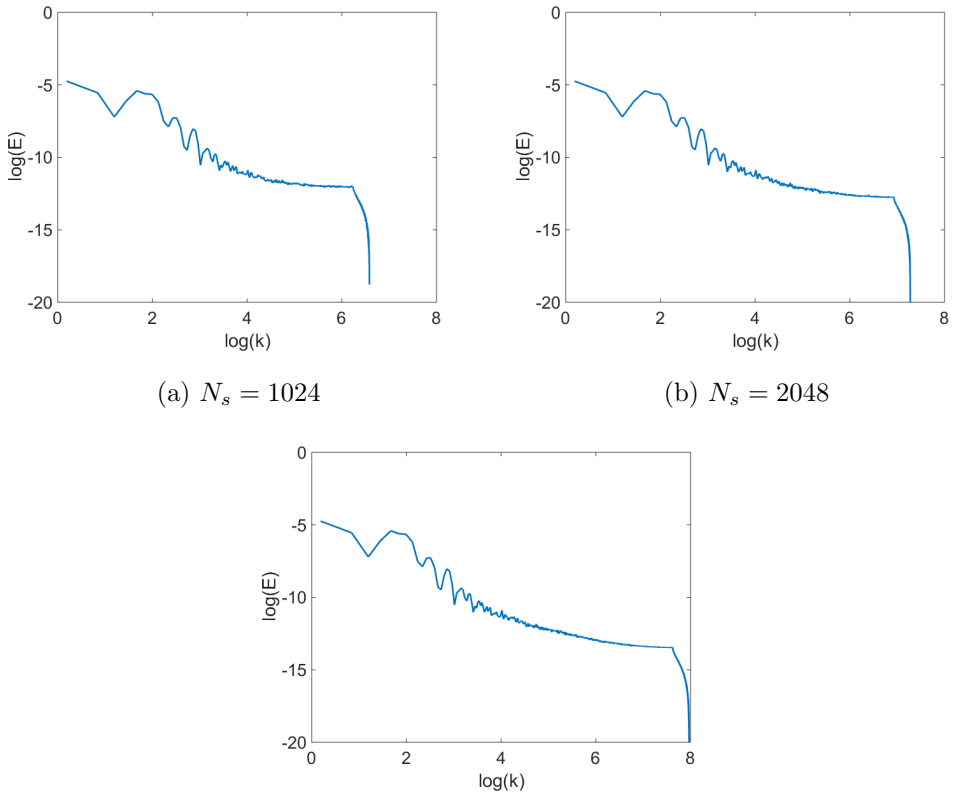
(d) 128× zoom (e) 512× zoom (f) 2048× zoom

(g) $8K\times$ zoom (h) $32K\times$ zoom (i) $128K\times$ zoom

Figure 4.4: Zooming in the solution at $t = 500$ seconds

For the solution to be physically admissible, we expect the energy of these features to decay exponentially with the length scales. We validate this by producing the energy spectrum plots calculated from a solution sampled on very fine grids. We vary the size of the sampling grid to check the consistency

of spectrum produced. On the log-log plot as shown in fig. 4.5 we observe that energy associated to high frequencies decays exponentially.



(a) $N_s = 1024$ (b) $N_s = 2048$

(c) $N_s = 4096$

Figure 4.5: Energy Spectrum computed on different Sampling grid size N_s

Enstrophy Preservation

Enstrophy is defined as the L^2 norm of the vorticity over the entire domain.

$$Z = \int_{\Omega} \omega^2(x,t)\, d\Omega \tag{4.7}$$

The solution of the incompressible Euler equations has the property to preserve its Enstrophy. The numerical solutions may obey this property with

some error. In this section we observe the evolution of enstrophy of the numerical solution along with time and its dependence on various parameters of the simulation.

Figure 4.6 shows the normalized enstrophy versus time plot of a simulation ran on a 128×128 grid.

$$\hat{Z}(t) \;=\; \frac{Z(t)}{Z(0)}, \tag{4.8}$$

We observe a 1.6% increase of Enstrophy by the end of the simulation.

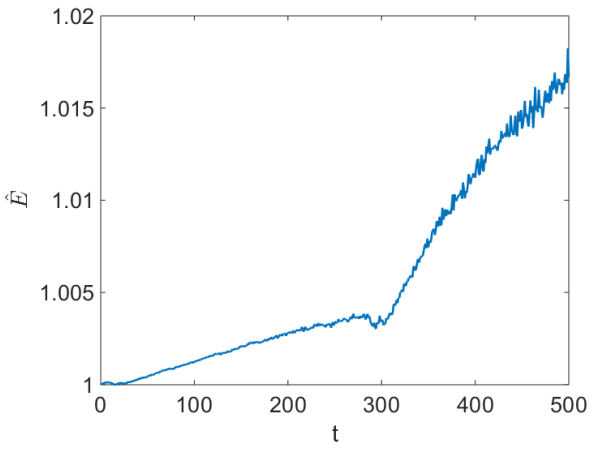
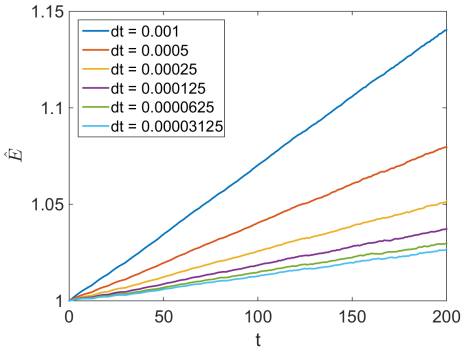


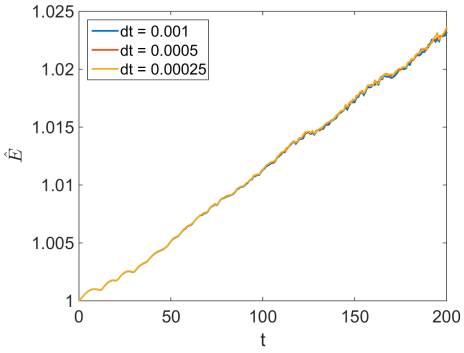
Figure 4.6: Normalized Enstrophy $\hat{Z}(t)$ on grid size 128×128

Upon varying the time step of the simulation we observe that this growth of enstrophy is effected only in the case where the temporal integration scheme in algorithm 1 is used. For the scheme in algorithm 2 time step plays no role in the observed increase of the enstrophy. This is shown in fig. 4.7.

Varying the grid size has clear influence on the enstrophy as shown in the figure 4.8 Plotting the overall increase of the normalized enstrophy shows a third order dependence on the grid size as shown in the figure 4.9.

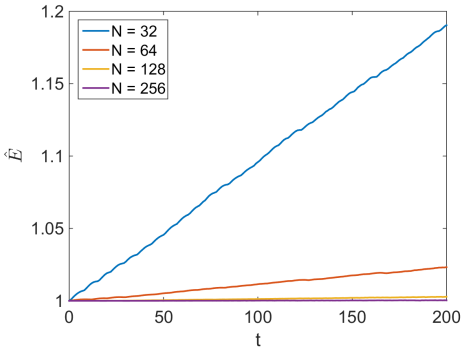


(a) 1-Stage temporal scheme (1)

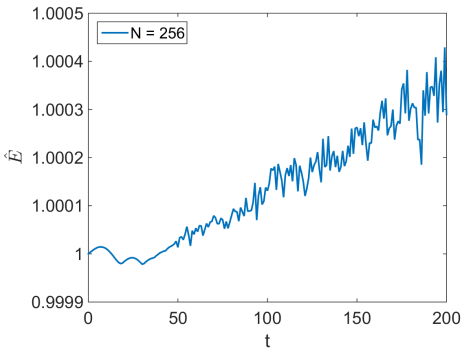


(b) 2-Stage temporal scheme (2)

Figure 4.7: Enstrophy for different time-step



(a) Comparison of Enstrophy evolution for different grid-size



(b) Enstrophy evolution for grid-size 256

Figure 4.8: Enstrophy for different grid-sizes

Time Reversibility

The system of incompressible Euler equations is time-reversible, meaning that the simulation can be ran backward by taking a negative time step. We validate the reversibility of the numerical scheme by running the simulation to a small time T, and then backwards. Figure 4.10 shows the solution for a simulation turned backwards at $T = 50$.

Taking the difference between the map at final and initial times $\vec{\chi}(:, 2T) -$

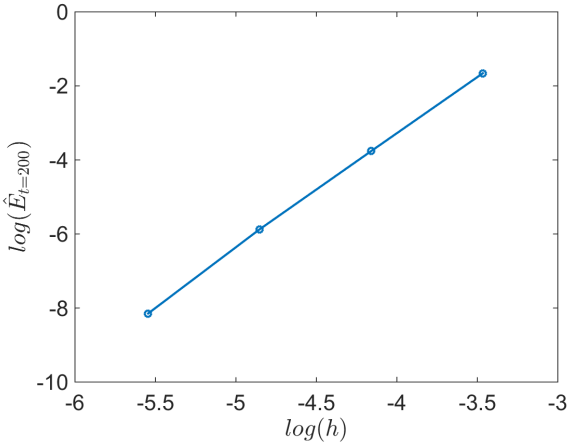
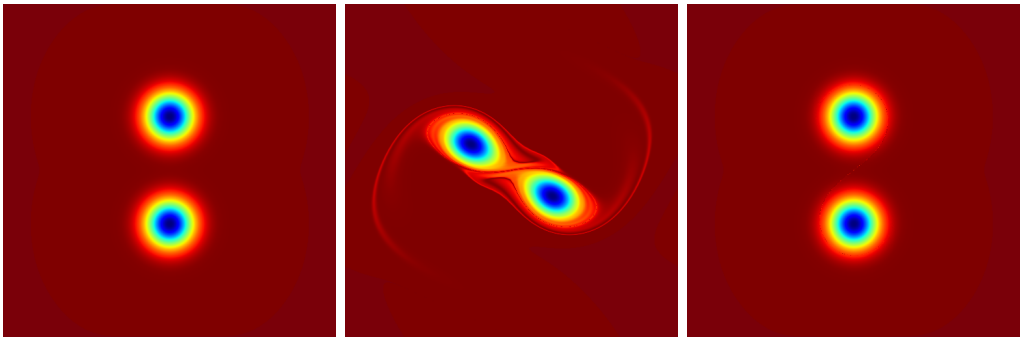


Figure 4.9: Enstrophy versus grid size h at $t = 200$



(a) t = 0 (b) t = T (c) t = 2T

Figure 4.10: Time Reversed simulation: $T = 50$ seconds

$\vec{\chi}(:, 0)$ gives us the numerical error of the computed solution. We compute this numerical error for different grid sizes and time steps, and observe a global third order accuracy of the method (fig. 4.11).

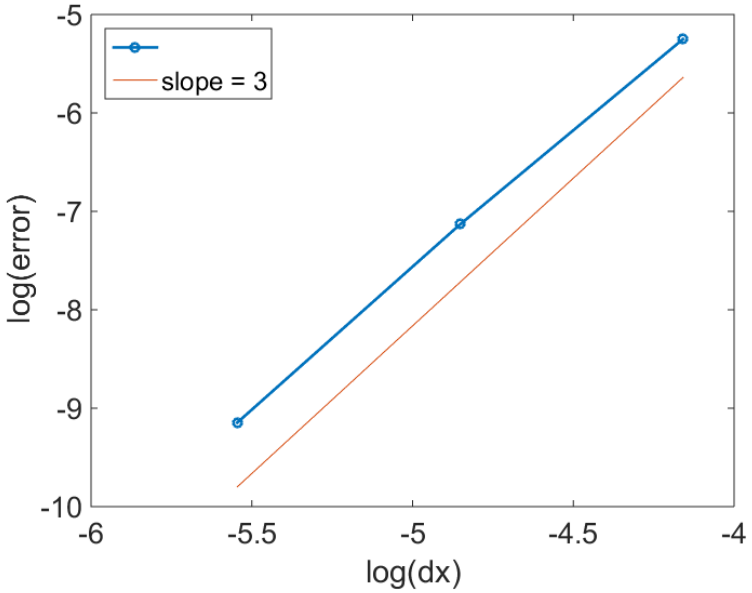


Figure 4.11: Log-log plot of error versus grid size ($dt \propto dx$)

4.3.2 Homogeneous Isotropic flow

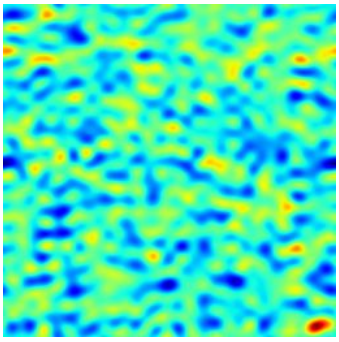
A fully developed homogeneous isotropic turbulence [14] exhibits a property of forming an energy cascade (transfer of energy from large to small length scales). Specific cases of the flow conditions achieves specific structure of the cascade mainly in terms of the slope ([10], [15]). The transfer of energy from larger to smaller length scales happens due to inertia and the mechanism is essentially inviscid. Therefore, the formation of energy cascade is also common in 2D incompressible Euler's equation [5].

We compare our results of homogeneous isotropic 2d-incompressible inviscid flow with the results established by Bachelor in 1960 [1] under the assumptions of vanishing enstrophy dissipation [18]. The numerical solution of vorticity transport equation has no enstrophy dissipation except the numerical diffusion caused by the scheme. Ideally, we expect an enstrophy spectrum of form k^{-1}.

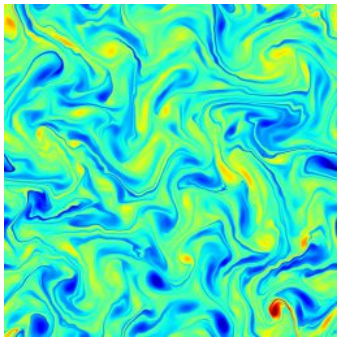
We initialize the simulation with a homogeneous isotropic incompressible flow field with energy spectrum peaking at a certain chosen length scale. We maintain the initial condition periodic and analytic by storing it in Fourier space as a superimposition of Fourier basis functions with complex amplitudes. Initial vorticity field comprises of numerous small vortices with maximum energy at the chosen length scale.

We run the simulation on a computational grid of size 128×128 with a time step $dt = \frac{1}{128}$ seconds. As time proceeds (fig. 4.12), the energy is transferred between the length scales forming a cascade as shown in fig. 4.13 and 4.14.

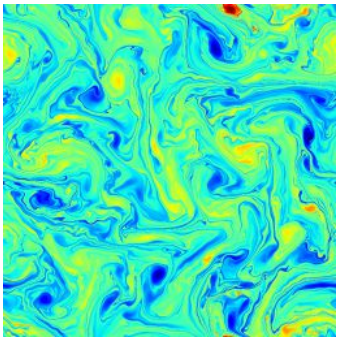
We observe a region in the enstrophy spectrum with the slope of -1. Some disturbances are seen at the smaller length scales which are expected to be caused due to the numerical diffusion caused by the scheme. This implies a promising initial validation of the scheme. Although, future work would involve a detailed analysis of the problem starting with a rigorous prediction of the expected spectrum.



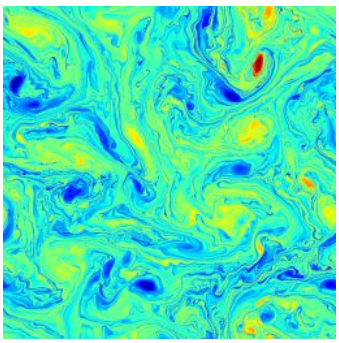
(a) $t = 0s$



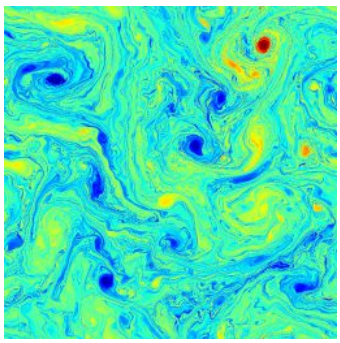
(b) $t = 500s$



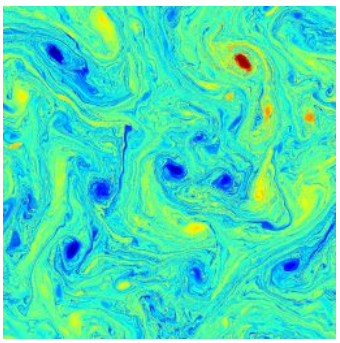
(c) $t = 1000s$



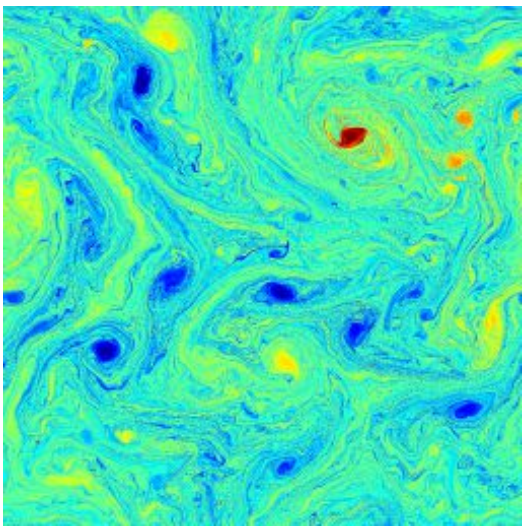
(d) $t = 1500s$



(e) $t = 2000s$



(f) $t = 2500s$



(g) $t = 3000s$

Figure 4.12: Homogenous Isotropic Flow

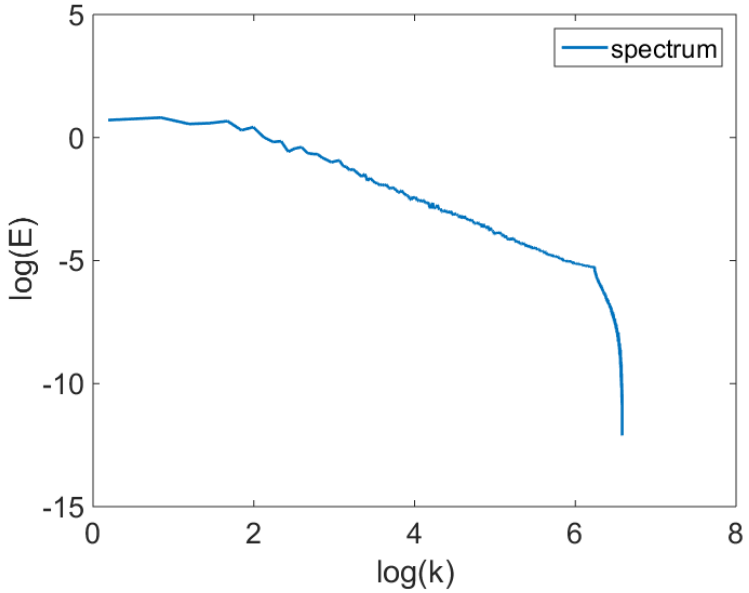


Figure 4.13: Time averaged energy spectrum at $t = 3000s$

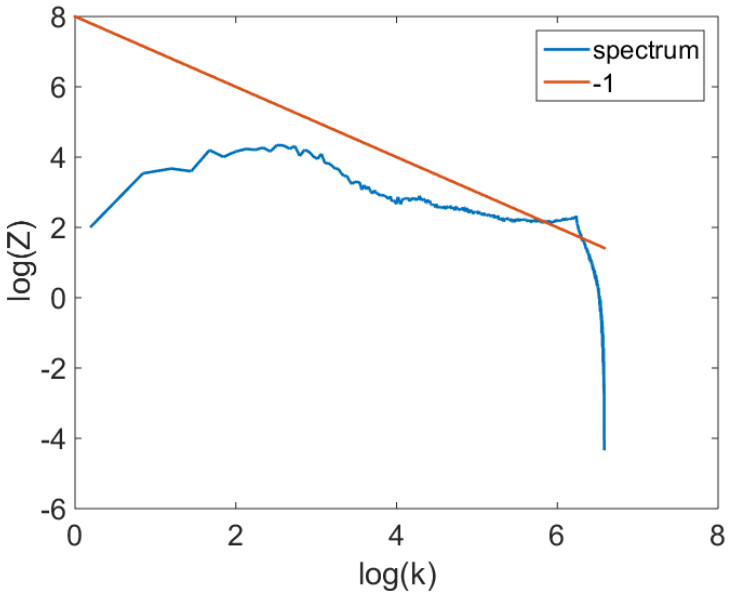


Figure 4.14: Time averaged enstrophy spectrum at $t = 3000s$

Chapter 5

Conclusions

The characteristic mapping method along with the map-stacking technique, can be used to solve non-linear advection problems rather advantageously. The solution could represent very fine sub-grid features having a very low contribution in the energy.

The method is compact, and thus the parallel implementation is robust with minimal inter-thread communication as opposed to in traditional higher-order methods which have larger stencils. The numerical method can be broken down to fast-Fourier-operations, Hermite interpolation and backward integration. All three of them could be very well optimized for parallel architecture.

Future work would involve a detailed analysis of the numerical scheme in regard of its accuracy and physical admissibility of the sub-grid features. Finally, the method as it is can be used in various computer-graphics applications, where fast computation of realistic fine features makes the applications very fast and robust.

Bibliography

[1] George K Batchelor. Computation of the energy spectrum in homogeneous two-dimensional turbulence. *Physics of Fluids (1958-1988)*, 12(12):II–233, 1969.

[2] RE Carlson and FN Fritsch. Monotone piecewise bicubic interpolation. *SIAM journal on numerical analysis*, 22(2):386–400, 1985.

[3] Prince Chidyagwai, Jean-Christophe Nave, Rodolfo Ruben Rosales, and Benjamin Seibold. A comparative study of the efficiency of jet schemes. *arXiv preprint arXiv:1104.0542*, 2011.

[4] Randall L Dougherty, Alan S Edelman, and James M Hyman. Nonnegativity-, monotonicity-, or convexity-preserving cubic and quintic hermite interpolation. *Mathematics of Computation*, 52(186):471–494, 1989.

[5] Gregory L Eyink. Dissipation in turbulent solutions of 2d euler equations. *Nonlinearity*, 14(4):787, 2001.

[6] L Ferracina and M Spijker. An extension and analysis of the shu-osher representation of runge-kutta methods. *Mathematics of computation*, 74(249):201–219, 2005.

[7] Morris R Flynn, Aslan R Kasimov, Jean-Christophe Nave, Rodolfo R Rosales, and Benjamin Seibold. On" jamitons," self-sustained nonlinear traffic waves. *arXiv preprint arXiv:0809.2828*, 2008.

[8] Frederick N Fritsch and Ralph E Carlson. Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, 17(2):238–246, 1980.

[9] James M Hyman. Accurate monotonicity preserving cubic interpolation. *SIAM Journal on Scientific and Statistical Computing*, 4(4):645–654, 1983.

[10] Robert H Kraichnan. The structure of isotropic turbulence at very high reynolds numbers. *Journal of Fluid Mechanics*, 5(04):497–543, 1959.

[11] Randall J. LeVeque. Numerical methods for conservation laws. 2002.

[12] Olivier Mercier and Jean-Christophe Nave. The characteristic mapping method for the linear advection of arbitrary sets. *arXiv preprint arXiv:1309.2731*, 2013.

[13] Jean-Christophe Nave, Rodolfo Ruben Rosales, and Benjamin Seibold. A gradient-augmented level set method with an optimally local, coherent advection scheme. *Journal of Computational Physics*, 229(10):3802–3827, 2010.

[14] Steven A Orszag and GS Patterson Jr. Numerical simulation of three-dimensional homogeneous isotropic turbulence. *Physical Review Letters*, 28(2):76, 1972.

[15] J Sommeria. Experimental study of the two-dimensional inverse energy cascade in a square box. *Journal of fluid mechanics*, 170:139–168, 1986.

[16] H Takewaki, A Nishiguchi, and T Yabe. Cubic interpolated pseudo-particle method (cip) for solving hyperbolic-type equations. *Journal of Computational Physics*, 61(2):261–268, 1985.

[17] Hideaki Takewaki and Takashi Yabe. Cubic-interpolated pseudo-particle (cip) method: Application to nonlinear or multi-dimensional problem. In *In Nagoya Univ. Proceedings of the US-Japan Workshop on Advanced Plasma Modeling 31 p (SEE N87-26627 20-75)*, volume 1, 1987.

[18] Chuong V Tran and David G Dritschel. Vanishing enstrophy dissipation in two-dimensional navier–stokes turbulence in the inviscid limit. *Journal of Fluid Mechanics*, 559:107–116, 2006.

[19] Bram van Leer. Towards the ultimate conservative difference scheme. *Springer Lecture Notes in Physics*, 18(2):163–168, 1973.

[20] Bram Van Leer. Towards the ultimate conservative difference scheme. ii. monotonicity and conservation combined in a second-order scheme. *Journal of computational physics*, 14(4):361–370, 1974.

[21] Bram Van Leer. Towards the ultimate conservative difference scheme iii. upstream-centered finite-difference schemes for ideal compressible flow. *Journal of Computational Physics*, 23(3):263–275, 1977.

[22] Bram Van Leer. Towards the ultimate conservative difference scheme. iv. a new approach to numerical convection. *Journal of computational physics*, 23(3):276–299, 1977.

[23] Bram Van Leer. Towards the ultimate conservative difference scheme. v. a second-order sequel to godunov's method. *Journal of computational Physics*, 32(1):101–136, 1979.