CLARKSON UNIVERSITY

Inverse Problems for Image Processing of Spatiotemporal Dynamical Systems

A Dissertation

By

Ranil Kumara Basnayake

Department of Mathematics

Submitted in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy, Mathematics

July 03, 2014

Accepted by the Graduate School

_____, _____

Date, Dean

Abstract

Optical flow is a classical problem in computer vision which was developed to determine the rigid body motion based on observed images. These concepts must be adapted for applications to other fields such as fluid mechanics and dynamical systems. Our approaches are based on an inverse problem formulation considering the physics of the observed density evolution and imposed prior knowledge of the solution. This presents itself in the form of a cost function which is a functional. The desired solution is a velocity field which is uncovered by minimizing the specifically designed functional in each individual scenario. This leads to a practical approach to analyze an observed dynamical system. Additionally we introduce a multi-frame version of the functional coupling of multiple images. Following the calculus of variations, this yields a coupled set of Euler-Lagrange PDEs which serve as an assimilation method that inputs video frames as driving terms. In this dissertation, we are especially interested in remotely observed fluid flows on a planetary scale such as oceanography as well as weather patterns here on Earth and on Jupiter. To overcome the coriolis effects on these systems, we introduce for the first time here a new method to reconstruct velocity fields incorporating quasi-static equations. Further, we illustrate the significance of these flow fields on analysis of mixing and mass transport in the fluid system being imaged.

Acknowledgments

This work is dedicated to those who believed in me, lead me to the best I could be and cheered me up during the tough times. I would personally like to thank all the supportive people around me, without whom it would not have been possible to accomplish this milestone.

First and the foremost, I would like to thank my advisor, Prof. Erik Bollt for making me a part of his research group. A simple email from him took me all the way to Clarkson and made me who I am today. I truly appreciate his guidance, insight, patience and all the opportunities he has given to me throughout these years.

I would like to thank Prof. Aaron Luttman, even though I missed his presence in Clarkson as my co-advisor. I'm extremely grateful for his advice, encouragement and support on an academic and personal level.

I would like to thank Professors Joseph Skufca, Scott Fulton, Daniel ben-Avraham and Jie Sun for serving as my thesis committee members and for their vision, comments and suggestions. Also I acknowledge the entire faculty of the Dept. of Mathematics for contributing to widen my subject knowledge.

I very much appreciate the financial support, received from the Office of Navel Research under grant N00014-09-1-0647 and the Department Mathematics at Clarkson University which allowed me to complete my Ph.D without any interruption.

I cherish all the memories I had with my "Chaos Lab Mates"; Sean, Ryan, Jiongxuan, Mike, Tian, Kelum, Linchen and Matt. I appreciate the true friendship and stress free work environment that my lab mates have provided. Special thanks goes to Matt, Tian and Dr. Cafero for helping me to finish the thesis and defense on time while thinking of their own work.

I thank from the bottom of my heart my parents for giving me freedom to make my own decisions while being supportive by all means. I'm thankful to my sisters and brother for their unconditional love and support. Also to my wife's family for strengthening me with love and care.

And finally, I must thank my wife for being with me to achieve this milestone scarifying her opportunities. Dinusha's support and encouragement made me succeed in my graduate studies. Also my son, Raveen and daughter, Ridmi for giving me a new meaning to my life. Their love simply refreshed my days and kept me going through the barriers.

The undersigned have examined the thesis/dissertation entitled **"Inverse Problems for Image Processing of Spatiotemporal Dynamical Systems "** presented by **Ranil Kumara Basnayake**, a candidate for the degree of **Doctor of Philosophy** (**Mathematics**), and hereby certify that it is worthy of acceptance.

Date	Erik Bollt (Advisor)
	Daniel ben-Avraham
	Scott Fulton
	Joseph D. Skufca
	Jie Sun

Contents

List of Tables

List of Figures

Chapter 1

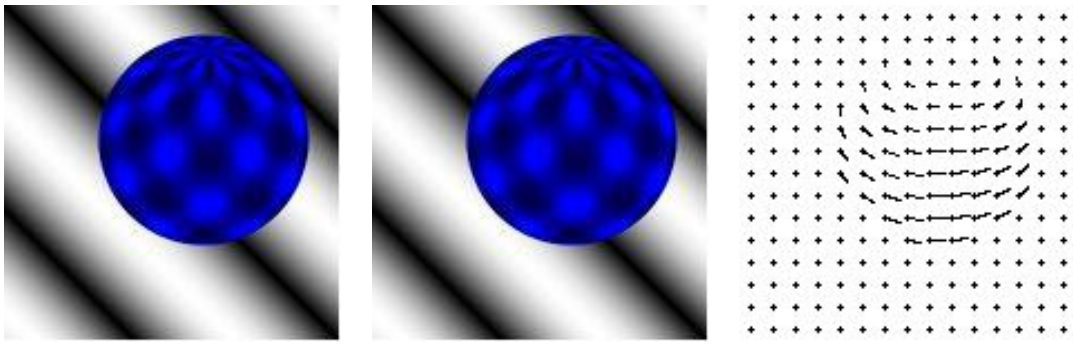
Introduction

A dynamical system is defined by a flow from a non-autonomous vector field $\dot{\mathbf{x}} = f(\mathbf{x}, t)$ that describes the evolution of the state $\mathbf{x}(t)$ at time t. We infer the vector field $f(\mathbf{x}, t)$ to determine and analyze the characteristics that govern the system. Recently, there has been a great deal of work analyzing in fluid systems such as ocean currents, cloud movements [1, 2] in scientific research. Analysis of fluids, a broad area which includes predictions, mixing and transport barriers, and the behavior of the motion of the fluid helps to understand the past, present and future behaviour of the system. Some analytical tools help to identify the hidden structures of systems which can not be seen by the human eye. Analysis of unsteady fluid flow dynamics requires non-autonomous velocity fields, however, the prior models to determine the velocity fields directly are not available for all such systems. One of the possible ways to get the information of these systems is to observe the system from a camera or a satellite and get a sequence of images during a particular time frame. In this work, our goal is to use such images of an observed system to approximate the velocity fields governing the motion of the system. These approximated velocity fields may be used to analyze the observed system with the help of analytical tools. Therefore, we mainly focus on analysis of unsteady fluid flow dynamics inferred using a sequence of image data of

the system, taken by a movie camera or even from a satellite.

Since the researchers have been successful with their techniques on systems like ocean currents and cloud movements, they expanded their interest to the fluid systems on other planets such as Jupiter [3–5]. In the process of analysis, most of the analytical tools such as Lagrangian Coherent Structures (LCS) [1, 6, 7] and coherent pairs [8] require the time-varying velocity fields of the system. However, in the absence of a prior model to determine the velocity fields of a fluid system, most of the analytical tools are inadequate. The technique we use to approximate the velocity fields is the optical flow computation which was introduced by Horn and Schunck [9] in 1981.

In the optical flow computation, the motion of a three-dimensional object is projected to a two-dimensional screen (image) and then two time-adjacent images of the scene are used to compute the motion field. The resulting flow field is a two-dimensional, representation of the apparent motion of the brightness patterns in the image. In Fig. 1.1, images (a) and (b) are two time-adjacent images of a rotating sphere which has different colors at neighboring points, and the image (c) shows the corresponding optical flow between images (a) and (b).



(a) (b) (c)

Figure 1.1: Rigid body motion – Images (a) and (b) show two consecutive time instances of the motion of a rotating sphere from right to left. The optical flow field between these two images is shown in image (c) [10].

The optical flow field and the motion field of a moving object are not always the same. For instance, consider a single color rotating sphere. Since there is no change in the brightness pattern between two images, the optical flow is zero even though there is motion. On the other hand, if we take two images of a still sphere in front of a moving light source, then the optical flow field is non-zero because the brightness pattern changes with the changes in illumination, but there is no actual motion. For convenience, researchers consider a simple system that assumes the motion field of an object can be directly identified by an optical flow field.

Determining the optical flow field involves solving inverse problems. If we have an initial image and the vector field is known, we can advect the first image using the known vector field to get the second image according to an appropriate model; this is called a forward problem. But in the optical flow computation, for two given images, we have to determine the governing motion field between those two images; that is an inverse problem.

The motivation of an optical flow computation is to determine vector fields either of a moving object or of an observed system, by considering the apparent motion between time-adjacent images of the same scene. Horn and Schunck [9] introduced the original optical flow algorithm to detect the motion field of a moving object. In their work, they made two assumptions. The first assumption, a brightness constraint, states that the image brightness of a point on the brightness pattern is constant over time for small motions. This includes a notion of rigid body motion. Then an energy functional is obtained measuring the errors of the brightness constraint over the image domain so that the velocity components u and v, along the x and y directions respectively, are obtained by minimizing the energy functional. In general, minimizing this energy functional is an ill-posed problem. Hence the above functional is regularized, see [11, 12], by making the second assumption, the smoothness constraint, that the expected flow is smooth. While the Horn and Schunck derivation was made in terms

of local considerations only, the same Partial Differential Equation (PDE) could be derived as a conservation law. The functionals obtained from the brightness constraint and the smoothness constraint are called the data term and the regularization term respectively. The total energy functional includes a regularization term to the data term with weighting factor α, which is called the regularization parameter.

After regularizing the energy functional, the flow components u and v can be reconstructed by minimizing the derived energy functional. We achieve this by choosing a suitable regularization parameter α [13–15] not only to balance the desire with the data term, but also to compromise with some form of regularity. We minimize the energy functional using a Calculus of Variations [16] approach. The Calculus of Variations was originally studied in the seventeenth century in Europe in the investigation of a number of mechanical and physical problems such as the work of Fermat on geometrical optics (1662) and the problem of Newton (1685) for the study of bodies moving in fluids [17]. However the brachistochrone problem proposed by John Bernoulli in 1696 played a significant role in the development of the Calculus of Variations. The Greek word *brachistochrone* means minimal time. The brachistochrone problem solved by John Bernoulli, James Bernoulli, Leibniz and Newton involved the determination of the minimum transit time of a particle between two fixed points along a wire under the influence of gravity. The resulting brachistochrone path was a cycloid [16].

Euler and Lagrange, who worked on a systematic way of dealing with problems in the Calculus of Variations, turned a new chapter by the first order necessary conditions to have an extremum for a functional. The Euler– Lagrange equation they introduced was a strong influence on future researchers, and their work was extended in many ways by Bolza, Hahn, Hamilton, Hilbert, Jacobi, Weierstrass and others [17]. Further, Euler – Lagrange equations appeared to be a convenient tool in minimizing functionals. We apply Euler-Lagrange equations to the optical flow energy functional,

as explained in Sec. 2.0.1. For this problem, we have two coupled Euler-Lagrange equations to be solved for u and v. The resulting PDE system generally allows for known and relatively simpler numerical techniques like the Gauss-Seidel method, the gradient descent method, or the LU factorization with Gaussian elimination to determine the solution.

While Horn and Schunck developing the optical flow algorithm to capture the motion of an object, Mitchell [18] introduced a method to compute vector fields in fluid systems. In this manual method, two-time adjacent images of an observed fluid motion are considered. First a special feature such as clouds is identified on first image then, the displacement of the special feature between two images is measured. Using the displacement and the time lapse between the two images, the velocity vector is defined at the center of the identified feature. However, the obtained vector fields are sparse, hence, they are inadequate for analytical tools as it is not possible to compute partial derivatives spatially of the vector field accurately [19]. Because of the sparsity of the velocity field and of errors which occur in the manual method when selecting features, automated algorithms such as Particle Image Velocimetry (PIV) [20, 21] and Correlation Image Velocimetry (CIV) [22] have been developed to determine the vector fields of the fluid flows. The basic experiment setup for PIV method is set by seeding the flow with small particles. Then the velocity of the tracer particles is computed. It is assumed that the flow and the tracer particles have the same velocity field. In the computation of velocity fields, correlations between two (or more) images is obtained. These automatic methods are capable of computing large numbers of velocity vectors compared to the manual method. However, the correlation between two images is obtained only if the time between the two images is less than 50 minutes for motions like Jupiter's atmospheric motion. Therefore, a new method, Advected Correction Correlation Image Velocimetry(ACCIV) [19] was developed to determine the velocity fields between two images which more than two

hours apart. In the ACCIV method, the authors use CIV as a subroutine to compute the velocity fields. Then, in the second step, they advect each point in the first image forward in time and the second image backward in time using the computed velocity. If an advected point does not match with its tie-point in the other image, a new path is obtained by interpolating two mismatched paths. In this way, they make a correction for the velocity vector by developing a new vector for the corresponding point by interpolation.

However, the above image-based methods can only be employed under rigid experimental settings. Therefore, to achieve the same goal as the image-based methods, simple optical flow algorithms have been developed for computation of geophysical fluid flows [23, 24], atmospheric motion [25] and laboratory experimental fluid flows [26]. Since the optical flow-based techniques may be applied in more general settings and use simpler algorithms, there is a tendency to develop optical flow-based approaches to determine the fluid flows. For instance, the authors in [27] have analyzed the dynamics of species transport in the ocean and we have analyzed the transport barriers for the sea surface temperature [28] by employing optical flow methods on satellite images without any experimental setup.

As we already know, the Horn and Schunck optical flow formulation with its assumptions of conservation of image intensity and smoothness of velocity field was developed for computing flow fields of rigid-body motions with non-reflectant surfaces. However, later researchers have incorporated different physics into the energy functional by introducing new data terms and the regularization terms. In most of the above cases, researchers introduced new terms to adapt the optical flow algorithm from rigid-body motion to fluid motion. The authors in [29–33] proposed an important assumption that a fluid system is evolving according to the continuity equation rather than according to the conservation of energy. In addition to the data term, they also proposed a new regularization term using the "second-order div-curl" in [34]. Af-

ter that the authors in [35] reformulated the optical flow problem with "second-order div-curl" regularization to capture the curl-free and divergence-free components. This was a big step forward from classical optical flow problems to the fluidic optical flow algorithms.

The developments in the optical flow methods for fluid motions motivated us to develop our own method to capture the motion of a fluid system. In our first approach in the field of optical flow, we developed an algorithm to compute velocity fields of a fluid system [36, 37] using observed images. This work is motivated by the work in [29–33], but in our approach, we represent the flow as a gradient of a potential function or the symplectic gradient of a stream function. In this way we need to determine only one unknown which is either a potential function or a stream function. Rather than reconstructing the horizontal and vertical components of the flow separately, this new approach allow us to reconstruct only one potential function or a stream function and then derive the horizontal and vertical components of the flow field. However, it is not always possible to represent large-scale fluid flows, such as oceanic flows and atmospheric flows from a gradient of a single potential function or a symplectic gradient of a stream function. In that case, local features such as vortices, saddle points and sources which are available very often in fluid flows need be modeled. We have applied our algorithm to these kinds of local structures, and the corresponding results are promising. Our method is very simple, simpler than that used in [35], and hence it is easy to implement the algorithm computationally. Another advantage of our method is we need to solve only one problem rather than two coupled problems as in other methods. More importantly the algorithm allows us to impose scientific priors via regularization directly on the potential or stream function rather than on the separate flow components u and v .

To check the accuracy of the algorithms we develop, we generated three benchmark data sets which represent a source, a gyre and a hyperbolic fixed point. We

constructed these data sets by evolving an initial condition according to the continuity equation with known velocity fields. A detailed explanation and the data sets are available in Sec. 2.3. We will demonstrate stream function formulation on these three benchmark data sets, and also the sea surface temperature (SST) data from the coast of Oregon, U.S.A. as explained in Sec. 3.3.1.

Another way to extend the optical flow computations is to introduce new numerical approaches to improve the convergence of the flow-computation algorithms. In [36, 38], it is shown that rather than reconstructing the components $\langle u, v \rangle$ of the flow, it is possible to reconstruct the stream function associated with the flow instead when the flow is the symplectic gradient of a stream function. For either case of reconstructing the velocity components or the associated stream function, the optical flow functional can be regularized with the total variation of the flow components or the total variation of the stream function, in order to more accurately capture the dynamical structures in the flow. It is well known that regularizing via the total variation results in signals that are approximately piecewise constant. In the case of regularizing with the total variation of the flow components, a flow whose components are piecewise constant will be favored. This is especially appropriate for laminar flows or flows that are approximately parallel to the coordinate axes. In the case of regularizing with the total variation of the stream function, the stream will be piecewise constant. This is especially appropriate for sparse flows. In this work we show that Total Variation (TV) regularization also leads to excellent reconstructions of other flow structures such as vortices, hyperbolic fixed points and sources in both the component-based and stream function formulations.

In order to minimize the TV-regularized optical flow functional, we linearize the associated Euler-Lagrange equations by adapting the Lagged Diffusivity Fixed Point Iteration (LDFPI) from [39]. In Sec. 5.1, we present the TV optical flow formulation and the LDFPI method. This is followed by results for synthetic data in Sec. 2.3

representing the dynamical structures of greatest interest for fluid flows, as well as to data generated from a data-driven ocean model of sea surface temperature data off the coast of Oregon, U.S.A.

This method has been extended in many directions, including the introduction of new data fidelities [29, 40] and regularization terms [34, 41], as well as on numerical methods to enhance the accuracy and convergence of the corresponding algorithms [42]. In the case of imaging fluid dynamics, the flow is generally not smooth, and the the turbulent structures are those of greatest interest. In addition to the above, we extended the optical flow computation for more than one time step to incorporate some temporal regularity to the energy functional.

Figure 1.2: Hurricane Sandy – Image shows one time instance of Hurricane Sandy in November 29 in 2012 [43].

Our main goal here is to analyze unsteady fluid flow dynamics using a sequence of image data of the system taken by a movie camera or even from a satellite. For instance, suppose we have to deal with systems like Hurricane Sandy and the storms near Jupiter's great red spot as shown in Fig. 1.2 and Fig. 1.3. Previous methods

Figure 1.3: Jupiter data – This image was taken by Voyager 2 space craft which shows the violent storms in the region of Jupiter extending from the equator to the southern polar latitudes in the neighborhood of the Great Red Spot [44].

required that there were only small changes of scene between each image, but otherwise the optimization approach just described yields a spatially regularized vector field in as far as the regularity term in the cost function is emphasized. However, if the scene in the movie data changes significantly between frames of the movie due to a relatively fast changing non-autonomous system, then there can be undesirable irregularity in the inferred vector field. This motivated us to develop a new approach to emphasize temporal regularity. The new approach of computing optical flow uses multiple images rather than just two. We call this approach a *multi-time step optical flow* for use when a sequence of images is available. For the computation of one stream function at a time, the multi-time step method and the stream function methods are the same. However, when we compute more than one stream function for consecutive time points of the system, we wish to adjust the functional simultaneously to emphasize that the stream functions of two consecutive time adjacent motions have similar behavior. This new assumption would incorporate an additional term in the energy

functional with a weighting factor β.

We will demonstrate the multi-time step method on two benchmark data sets representing a gyre and a source as introduce in Sec. 2.3, and also the sea surface temperature (SST) data from in Sec. 3.3.1. Furthermore, we will employ our method on a planetary data which represents the Jupiter's atmospheric motion near the Great Red Spot as introduce in Sec. 6.2.2.

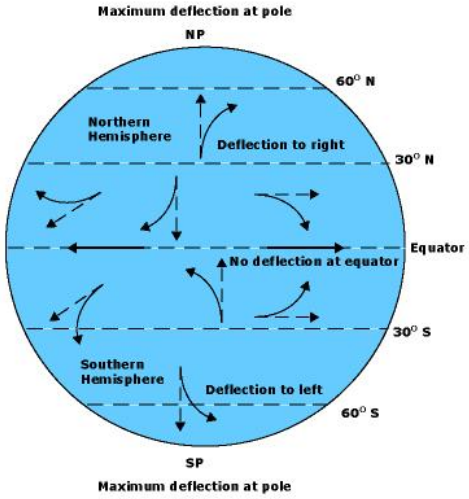
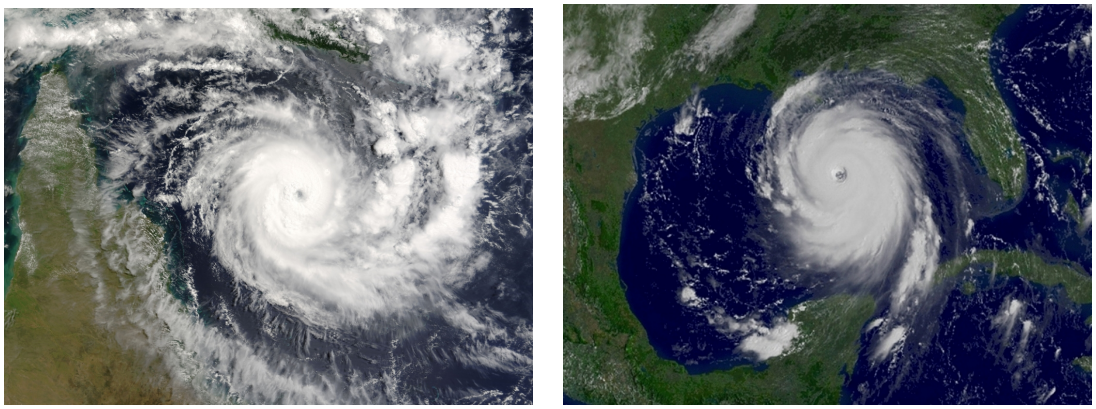


Figure 1.4: The coriolis effect – The image shows the coriolis effect in the moving objects on the earth. In northern hemisphere deviation is to the right and in southern hemisphere the deviation is to the left [45].

Based on the accuracy of the results we obtained after performing our algorithms on synthetic data, the next step was to apply our algorithms to real data sets such as Jupiter's storms in Fig. 1.3. Since planets are rotating about their axes, the coriolis force may affect the above systems. As an example of a real world scenario for the coriolis effect, we can consider two storms that occurred few years ago. The image (a) in Fig. 1.5 shows a one-time instance of Cyclone Ingrid over northern Australia. Wind flows from high pressure to low pressure and the coriolis effect create the appearance of the motion as counter-clockwise swirls. The counter-clockwise swirls occur because

the storm is in southern hemisphere. On the other hand, in image (b), the wind swirls clockwise direction in the northern hemisphere. The image represents Hurricane Katrina above the Gulf of Mexico [46].



(a) (b)

Figure 1.5: Coriolis effect in nature – Image (a) shows the Cyclone Ingrid stormed over the northern Australia. Since this is in the southern hemisphere, the storm swirl in counter-clockwise direction. The image (b) shows the Hurricane Katrina above the Gulf of Mexico. The storm swirls in clockwise direction since this is in the northern hemisphere [46].

Due to the coriolis force, there may be an apparent deflection of the path of an object that moves on a rotating coordinate system though the object does not actually deviate. Fig. 1.4 illustrates the deflection clearly. As shown in Fig. 1.5, due to earth's rotation, global wind and ocean currents in the northern hemisphere deviate to the right relative to the direction of motion and wind and currents in the southern hemisphere deviate to the left. Therefore, when we deal with satellite images of systems affected by coriolis force, we have to incorporate the coriolis effects in the energy functional to determine the optical flow fields accurately. To achieve this, we developed a new optical flow algorithm that incorporate with the coriolis force. In this case we included quasi-static equations [47] in the energy functional and then used the usual minimization process to determine the solution.

We first demonstrate the quasi-static optical flow method, using a benchmark data set which is affected by the coriolis force as generates in Sec. 7.4. Then we apply the method on a data set of Jupiter's atmospheric motions which we introduced in Sec. 7.5. Then we developed multi-time step method for quasi-static algorithm to improve the accuracy of the results. We also verify the accuracy of the computed flow fields for Jupiter data by advecting the points inside and outside the Great Red Spot using an artificial boundary.

Figure 1.6: Sacramento River – The image shows two branches of Sacramento river meeting at this point. After they meet, a natural barrier forms between murky water and the clear water [48].

Once we compute the vector fields for an observed system, the next task is to analyze the system using the computed vector fields. In this thesis, we analyze the systems by determining the Lagrangian coherent structures. Lagrangian coherent structures form barriers between two dynamically distinct systems. As an natural example, Fig. 1.6 shows two branches of the Sacramento River meet at this intersection point. One branch carries clear water and the other branch carries murky water.

However, even after they meet, a mixing barrier forms between the clear water and the murky water. This barrier can be treated as a natural Lagrangian Coherent Structure.

Finally, we will demonstrate the use of the computed vector field to analyze mixing and mass transport in the fluid system being imaged. Several methods such as determining Lagrangian Coherent Structures (LCSs) [1,6,7] and coherent pairs [8] are available to achieve this goal. In this endeavor we compute Finite Time Lyapunov Exponents (FTLE) at each and every point in the system to determine LCSs. In the computation of FTLEs, we consider two nearby points at time t_0 and measure the separation of the trajectories over the time period $[t_0, T]$. If the separation is relatively high, the set of corresponding points are barriers for mixing and mass transport in the fluid system. These separation barriers on the FTLE fields are the LCSs. The LCS for the SST data set are computed using the vector fields obtained from the multi-time step method and shown in Fig. 8.12. We complete our discussion by computing the LCSs for the Jupiter's atmospheric motion near the Great Red Spot using the computed flow from the quasi-static optical flow method.

Chapter 2

Classical Optical Flow Method

According to the original Horn and Schunck formulation of optical flow [9], the image brightness $I(x, y, t)$ at a point (x, y) is assumed to be locally conserved over time if the motion is small.

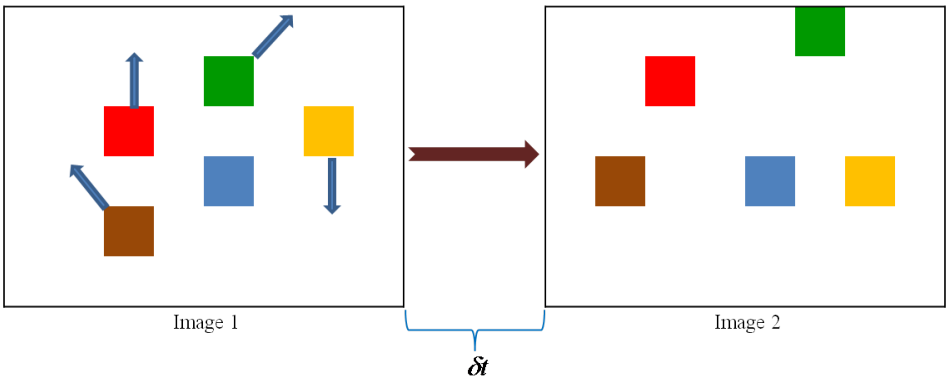


Figure 2.1: Intensity changes – Images 1 and 2 show two time adjacent images of an observed system which is not still. Five selected points are shown in image 1. In image 2, whether those points are moving or not, the color of those points does not change.

The Fig. 2.1 demonstrates the motion of five selected points in two time instances of an observed system. These two images are δt apart with respect to time. Among the five points, four are moving and one (blue color) is still, but the color of all five

points remains unchanged with respect to the motion. This implies that the image brightness is locally conserved. Now consider a point (x, y) on the image 1 and let u and v be the velocity components along the x and y directions respectively. Then the displacement $(\delta x, \delta y)$ of the point (x, y) after δt time is given by $(x + u\delta t, y + v\delta t)$. The first image in Fig. 2.2 shows a specific point (x, y) which is green in color moving to a new point $(x + u\delta t, y + v\delta t)$ in the second image, but the color of the point (green) does not change.

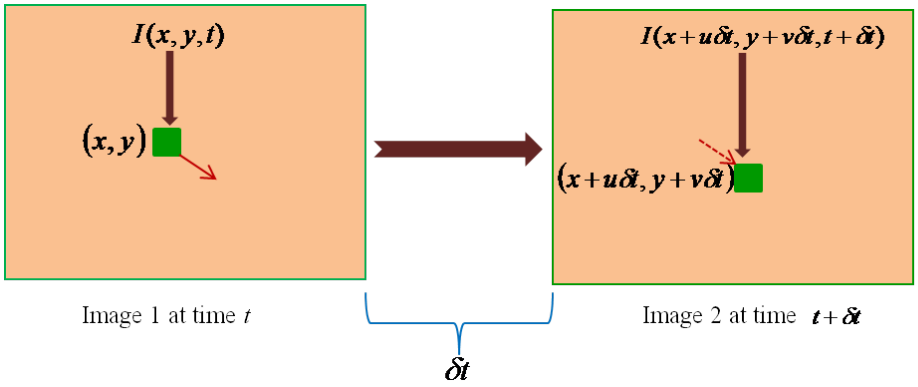


Figure 2.2: Conservation of image intensity – Image 1 and image 2 represent two time adjacent images of a moving system. The point (x, y) in the image 1 moves to the point $(x + u\delta t, y + v\delta t)$ in image 2, but the the color does not change.

This leads us to make the first assumption that the image brightness of a point (patch) in the pattern does not change over time; hence we can express the relationship with the unit time interval $\delta t = 1$ as,

$$I(x + u, y + v, t + 1) = I(x, y, t). \tag{2.1}$$

Now, further assuming the motion between two images is small and taking the first order Taylor series expansion of the left hand side of Eq.(2.1) about (u, v, t)

$$I(x, y, t) + (I_t + I_x u + I_y v) = I(x, y, t), \tag{2.2}$$

where it is assumed that the higher order terms in Taylor series expansion are equal to zero. Simplifying Eq.(2.2), we have

$$I_t + I_x u + I_y v = 0, \tag{2.3}$$

where I_t, I_x and I_y are partial derivatives of I with respect to t, x and y respectively. Note that $I(x, y, t)$ is the assumed data function that represents the gray scale color intensity of a point (x, y) on the scene domain Ω at time t. That is, $I : \Omega \times \mathbb{R}^+ \to \mathbb{R}$. Generally, actual data from a digital movie camera is pixelated at discrete spatial locations $\{x_i, y_j\}_{i,j=1}^{p,q}$ at times t_k as a complete data set $\{I_{i,j,k}\}_{i,j,k=1}^{p,q,T}$ at times t_k, where $t_1 = 0$ and t_k is the time after $k-1$ units. In other words, the first image was taken at $t = 0$ and the image k was taken after $k-1$ time units from when image 1 has been taken. Here p and q are the number of rows and the number of columns of the input images. This quantization may accumulate some errors and the measurements we obtained from real scenarios may come with some amount of noise. Therefore, the functional (data term) to minimize the errors can be obtained by integrating the errors of the brightness constraint over the image domain as

$$E_b(u, v) = \int_\Omega (I_t + I_x u + I_y v)^2 \, d\Omega. \tag{2.4}$$

To develop u and v, as suggested by the brightness constraint objective in Eq.(2.3), we state the functional in Eq.(2.4), however, minimizing $E_b(u, v)$ in Eq.(2.4) is an ill-posed problem.

Definition 1. *[49] Let $K : \mathcal{H}_1 \to \mathcal{H}_2$. An operator equation*

$$Kf = g \tag{2.5}$$

is said to be well-posed provided

1. *for each $g \in \mathcal{H}_2$ there exists $f \in \mathcal{H}_1$, called a solution, for which Eq.(2.5) holds;*
2. *the solution f is unique; and*
3. *the solution is stable with respect to perturbations in g. This means that if $Kf_* = g_*$ and $Kf = g$, then $f \to f_*$ whenever $g \to g_*$.*

A problem that is not well-posed is said to be ill-posed.

To avoid the ill-posedness of this problem, the data term must be regularized. We approach the regularization of this ill-posed problem as explained in Chapter 4 by assuming the expected flow is smooth. This implies that the partial derivatives of the velocity components u and v exist and hence the regularization term becomes

$$R(u,v) = \int_\Omega \left(u_x^2 + u_y^2 + v_x^2 + v_y^2\right)\, d\Omega. \tag{2.6}$$

Now the problem can be reformulated in terms of an energy functional obtained by combining the data term and the regularization term by weighting the second term with a non-negative regularization parameter α. Then the total energy functional to be minimized is given by

$$E(u,v) = \int_\Omega \left(I_t + I_x u + I_y v\right)^2\, d\Omega + \alpha \int_\Omega \left(u_x^2 + u_y^2 + v_x^2 + v_y^2\right)\, d\Omega. \tag{2.7}$$

Here the selection of a suitable regularization parameter is an important step and we will discuss it in Sec. 4.6 separately. In the process of minimizing the functional in Eq.(2.7), we need some theoretical background in Calculus of Variations. Therefore, we include the next section to illustrate how to minimize a functional through Calculus of Variations.

2.0.1 Euler-Lagrange Equations

To minimize the functional in Eq.(2.7), we will use a Calculus of Variations approach. In this subsection, we will demonstrate methods to minimize our functionals as explained in [16]. We first consider minimization of a simple functional and necessary conditions for the functional to have a minimum.

Suppose we are given a functional $J(u)$ and we must determine the optimizer $u(x) = \hat{u}(x)$ over the interval $a \leq x \leq b$

$$J(u) = \int_a^b F(x, u, u_x)\, dx. \tag{2.8}$$

Here $F(x, u, u_x)$ is a function with continuous first and second partial derivatives with respect to all of its arguments. Also, let $u(x)$ be a continuously differentiable function on $[a, b]$ which satisfies the boundary conditions

$$u(a) = A \qquad \text{and} \qquad u(b) = B. \tag{2.9}$$

Optimization necessitates that the first variations are stationary. Analogous to the first derivative of a function, we obtain the first variation of the given functional. First we give an increment $h(x)$ to the function $u(x)$ with the boundary conditions

$$h(a) = h(b) = 0. \tag{2.10}$$

The corresponding increment ΔJ in Eq.(2.8) with respect to the increment of $h(x)$ is

$$\Delta J(u) = \int_a^b [F(x, u + h, u_x + h_x) - F(x, u, u_x)]\, dx. \tag{2.11}$$

Using the second order Taylor series expansion, we can expand the integrand of the functional in Eq.(2.11) as

$$\Delta J(u) = \int_a^b [F_u h + F_{u_x} h_x]\, dx + \frac{1}{2}\int_a^b [F_{uu} h^2 + 2F_{uu_x} h h_x + F_{u_x u_x} h_x^2]\, dx + \epsilon, \quad (2.12)$$

where ϵ represents the higher order terms. The first variation of the functional in Eq.(2.8) with respect to the argument variable u is defined as the integral of first order terms in Eq.(2.12) and is given by

$$\delta J(u) = \int_a^b [F_u h + F_{u_x} h_x]\, dx. \quad (2.13)$$

Just as we set the first derivative equal to zero when determining optimizers, the first variation is set to be zero to find the optimizers of the functional. The following theorem explains a necessary condition to determine the optimizers of a functional.

Theorem 2.0.1. [16] *A necessary condition for the differentiable functional $J(u)$ to have an extremum for $u = \hat{u}$ is that its variation vanish for $u = \hat{u}$, i.e., that*

$$\delta J(h) = 0 \quad (2.14)$$

for $u = \hat{u}$ and all admissible h.

From the above theorem, the necessary condition, called the Euler-Lagrange equation, for the functional in Eq.(2.8) to have an extrema can be obtained as

$$\frac{\partial F}{\partial u} - \frac{d}{dx}\left(\frac{\partial F}{\partial u_x}\right) = 0. \quad (2.15)$$

By solving the Euler-Lagrange equation in Eq.(2.15), the optimum argument variable $\hat{u}$ which maximizes or minimizes the functional in Eq.(2.8) can be determined. However, an additional condition is required to determine whether the optimal function $\hat{u}$

maximizes or minimizes the functional. The new condition is obtained by considering the second variation of the functional called the Legendre's condition. By definition, the second variation of the functional in Eq.(2.8) with respect to the variable u is the integral with second order terms in Eq.(2.12). Therefore, the second variation is obtained as

$$\delta^2 J(u) = \int_a^b [F_{uu}h^2 + 2F_{uu_x}hh_x + F_{u_xu_x}h_x^2]\, dx. \tag{2.16}$$

In a similar way of determining whether the optimizers of a function are minimizers or maximizers using the second derivative test, the sign of the second variation of a functional is considered to determine whether the optimizer is a minimum or maximum. The following theorem explains a second variation test for functionals.

Theorem 2.0.2. [16] *A necessary condition for the functional $J(u)$ to have a minimum for $u = \hat{u}$ is that*

$$\delta^2 J(u) \geq 0 \tag{2.17}$$

for $u = \hat{u}$ and all admissible h. For a maximum, the sign $\geq$ in Eq.(2.17) is replaced by $\leq$.

Testing the Legendre's condition for optical flow functionals is not an easy task. In the optical flow computation however, we always have to minimize an energy functional to reconstruct the velocity components. Therefore, considering the difficulty of comparing the sign of the second variation and the expectation that we always have a minimizer, we do not use the above theorem in the rest of our discussion. Instead of the Legendre's condition, we use an alternative approach to show that a minimizer exists. Further discussion of this approach is available in Sec. 2.1. In this way, we only need the Euler-Lagrange equation to determine the minimizers of the functional. Therefore, the next step is to develop the Euler-Lagrange equation for functionals with more than one variable.

The relationship in Eq.(2.15) can be expanded to allow for many variables so we

are interested in expanding the results to a functional of the form

$$J(u,v) = \int_{\Omega} F(x,y,u,v,u_x,u_y,v_x,v_y)\, d\Omega, \tag{2.18}$$

allowing for vector fields $\langle u(x,y), v(x,y)\rangle$ in the plane $(x,y) \in \mathbb{R}^2$. Since there are two argument functions u and v, we may have two coupled Euler-Lagrange equations. The Euler-Lagrange equations for the functional in Eq.(2.18) follow as

$$\begin{aligned} \frac{\partial F}{\partial u} - \frac{\partial}{\partial x}\left(\frac{\partial F}{\partial u_x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial F}{\partial u_y}\right) = 0 \\ \frac{\partial F}{\partial v} - \frac{\partial}{\partial x}\left(\frac{\partial F}{\partial v_x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial F}{\partial v_y}\right) = 0. \end{aligned} \tag{2.19}$$

The above Euler-Lagrange equations are specialized below for functionals of the form in Eq.(2.7) and hence we can determine the velocity components by solving the resulting Euler-Lagrange PDE equations for u and v.

Another way to compute the Euler-Lagrange equations for a functional $J(u)$ in Eq.(2.8) is to compute the Gateaux derivative $J(u)$ with respect to u and set it to zero. The Gateaux derivative of the functional $J(u)$ is obtained as

$$DJ(u) = \frac{d}{d\tau} J(u + \tau h)|_{\tau=0} \tag{2.20}$$

for all admissible h.

2.1 Existence and Uniqueness of the Solution

In general, when we minimize the optical flow problem, assuming existence and uniqueness of the solution, we determine the minima by solving the Euler-Lagrange equations for each argument variable. The issues of existence and uniqueness of solutions of the PDEs in Eq.(2.19) follow the theory of convex optimization including

discussion of convexity, coercivity and lower semi-continuity of the specific functional $J(u, v)$.

Let $\mathcal{H}$ be a Hilbert space and $\mathcal{C}$ be a closed, convex subset of $\mathcal{H}$ and $J : \mathcal{H} \to \mathbb{R}$. Note that, we assume that $\mathcal{H}$ is a Hilbert space only for our convenience. The energy functionals which we are going to minimize should satisfy several conditions to guarantee that they have minimizers. Therefore, we now present the conditions that required for the existence and uniqueness of the solution as explained in [49].

Definition 2. *A sequence f_n in a Hilbert space $\mathcal{H}$ converges weakly to f_*, denoted by $f_n \rightharpoonup f_*$, if $\langle f_n, g\rangle \to \langle f_*, g\rangle$ for all $g \in \mathcal{H}$.*

Definition 3. *A functional $J : \mathcal{H} \to \mathbb{R}$ is weakly lower semicontinuous if*

$$J(f_*) \leq \liminf J(f_n)$$

whenever $f_n \rightharpoonup f_*$.

Definition 4. *A functional $J : \mathcal{H} \to \mathbb{R}$ is convex on $\mathcal{C} \subset \mathcal{H}$ if*

$$J(\tau f_1 + (1-\tau) f_2) \leq \tau J(f_1) + (1-\tau) J(f_2),$$

whenever $f_1, f_2 \in \mathcal{C}$ and $\tau \in (0, 1)$. Moreover, J is strictly convex provided the inequality is strict whenever $f_1 \neq f_2$.

Definition 5. *A functional $J : \mathcal{H} \to \mathbb{R}$ is coercive if*

$$J(f_n) \to \infty \quad \textit{whenever} \quad \| f_n \|_{\mathcal{H}} \to \infty.$$

Now the following theorem from [49] explains sufficient conditions for the functional $J(u, v)$ to have a minimum.

Theorem 2.1.1. [49] *Assume that $J : H \to \mathbb{R}$ is weakly lower semi-continuous and coercive and that C is a closed, convex subset of H. Then J has a minimizer over C. If furthermore, J is also strictly convex, then the minimizer is unique.*

When we reconstruct the velocity components by minimizing the functionals, we assume that there exists a solution for our functional and hence we only solve the Euler-Lagrange equations to determine velocity components.

2.2 Solution to the Optical Flow Problem

Recall that the energy functional in Eq.(2.7)

$$E(u,v) = \int_\Omega \left(I_t + I_x u + I_y v\right)^2 \, d\Omega + \alpha \int_\Omega \left(u_x^2 + u_y^2 + v_x^2 + v_y^2\right) \, d\Omega \tag{2.21}$$

which we minimize to reconstruct velocity components u and v. In the minimization process of this functional, we first apply the equations in Eq.(2.19) to the functional in Eq.(2.21) and then obtain the Euler-Lagrange equations as

$$\begin{aligned} I_x(I_t + I_x u + I_y v) + \alpha \nabla^2 u = 0 \\ I_y(I_t + I_x u + I_y v) + \alpha \nabla^2 v = 0. \end{aligned} \tag{2.22}$$

First we minimize the functional in Eq.(2.21) using the gradient descent approach which we often apply to optimize functionals. The gradient components of the energy functional with respect to u and v are given as

$$\begin{aligned} \delta u = I_x(I_t + I_x u + I_y v) + \alpha \nabla^2 u \\ \delta v = I_y(I_t + I_x u + I_y v) + \alpha \nabla^2 v. \end{aligned} \tag{2.23}$$

The gradient decent algorithm is an iterative method which updates u and v for given initial conditions u_0 and v_0 as

$$\begin{aligned} u^{(k+1)} &= u^{(k)} - \delta\tau[I_x(I_t + I_x u^{(k)} + I_y v^{(k)}) + \alpha\nabla^2 u^{(k)}] \\ v^{(k+1)} &= v^{(k)} - \delta\tau[I_y(I_t + I_x u^{(k)} + I_y v^{(k)}) + \alpha\nabla^2 v^{(k)}]. \end{aligned} \tag{2.24}$$

Here k represents the iteration number and $\delta\tau$ is the step size for each iteration where $\delta\tau$ should be sufficiently small to ensure the numerical stability. Recall $u^{(k)}$ and $v^{(k)}$ must be discretely represented on the grid $\{x_i, y_j\}_{i,j=1}^{p,q}$ and derivatives must be numerically approximated by finite differences.

On the other hand, we can solve the system in Eq.(2.22) for u and v using the Gauss-Seidel method by linearizing the system. Therefore, when we use the Gauss-Seidel method, first we approximate the Laplacian terms in Eq.(2.22). The approximations are given as

$$\begin{aligned} \nabla^2 u_{i,j,l} &\approx \kappa(\bar{u}_{i,j,l} - u_{i,j,l}) \\ \nabla^2 v_{i,j,l} &\approx \kappa(\bar{v}_{i,j,l} - v_{i,j,l}), \end{aligned} \tag{2.25}$$

where

$$\begin{aligned} \bar{u}_{i,j,l} &= \frac{1}{6}(u_{i-1,j,l} + u_{i,j+1,l} + u_{i+1,j,k} + u_{i,j-1,l}) \\ &\quad + \frac{1}{12}(u_{i-1,j-1,l} + u_{i-1,j+1,l} + u_{i+1,j+1,l} + u_{i+1,j-1,l}), \\ \bar{v}_{i,j,l} &= \frac{1}{6}(v_{i-1,j,l} + v_{i,j+1,l} + v_{i+1,j,l} + v_{i,j-1,l}) \\ &\quad + \frac{1}{12}(v_{i-1,j-1,l} + v_{i-1,j+1,l} + v_{i+1,j+1,l} + v_{i+1,j-1,l}) \end{aligned}$$

and $\kappa = 3$ for the Horn and Schunck method [9] when the average is taken according to the above approximations. Now substituting Eq.(2.25) in Eq.(2.22) and setting

them equal to zero, we have

$$\begin{aligned}(\alpha^2 + I_x^2)u + I_x I_y v &= (\alpha^2 \bar{u} - I_x I_t) \\ I_x I_y u + (\alpha^2 + I_y^2)v &= (\alpha^2 \bar{v} - I_y I_t)\end{aligned} \tag{2.26}$$

Solving the system for u and v, we have,

$$\begin{aligned}(\alpha^2 + I_x^2 + I_y^2)(u - \bar{u}) &= -I_x(I_x\bar{u} + I_y\bar{v} + I_t) \\ (\alpha^2 + I_x^2 + I_y^2)(v - \bar{v}) &= -I_y(I_x\bar{u} + I_y\bar{v} + I_t).\end{aligned} \tag{2.27}$$

To solve the problem using the Gauss-Seidel method, we rewrite the problem as

$$\begin{aligned}u^{k+1} &= \bar{u}^k - I_x(I_x\bar{u}^k + I_y\bar{v}^k + I_t)/(\alpha^2 + I_x^2 + I_y^2) \\ v^{k+1} &= \bar{v}^k - I_x(I_x\bar{u}^k + I_y\bar{v}^k + I_t)/(\alpha^2 + I_x^2 + I_y^2).\end{aligned} \tag{2.28}$$

For the given two time-adjacent images of a scene, we select the first image as I and then compute I_x and I_y using finite difference approximation. Also I_t is computed taking the difference between image 2 and image 1. Now for an initial conditions u^0 and v^0, each iteration updates u^{k+1} and v^{k+1} until they reach the solution.

2.3 Synthetic Data

In this section we present three synthetic data sets which represent three different kinds of dynamical structures that are important in fluid dynamics. For all three synthetic flows, the data (a sequence of images) was obtained by choosing an initial density $I(x, y)$ and evolving it forward in time according to the continuity equation

$$\frac{dI}{dt} = -(I_x u + I_y v + I u_x + I v_y) \tag{2.29}$$

using the vector fields governing each flow. The first example flow structure to be reconstructed is about a hyperbolic fixed point. The stream function corresponding to this data set is

$$\psi(x, y) = x^2 - y^2 \tag{2.30}$$

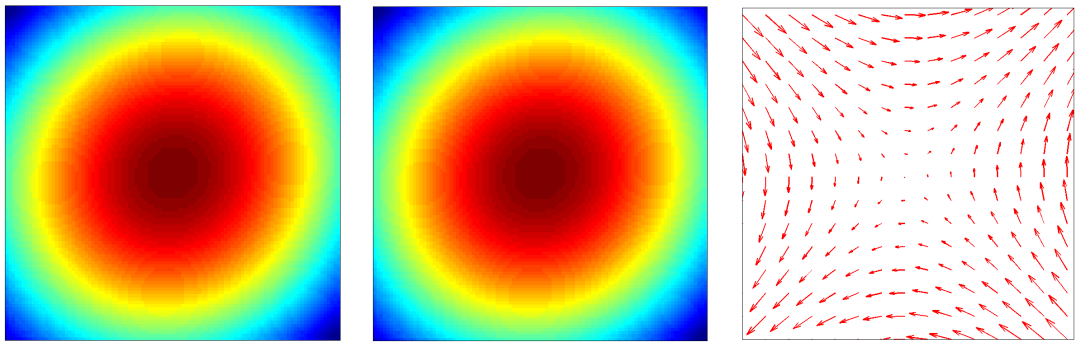
on the domain $[-0.5, 0.5] \times [-0.5, 0.5]$. Note that this velocity is not the gradient of a potential function, but it is the symplectic gradient of a stream function, i.e.

$$\langle u, v \rangle = \langle -\psi_y, \psi_x \rangle. \tag{2.31}$$

Then the velocity components which represent the hyperbolic fixed point are obtained as

$$\langle u, v \rangle = \langle 2y, 2x \rangle \tag{2.32}$$

on the domain $[-0.5, 0.5] \times [-0.5, 0.5]$. Two time instances after a few iterations of the density evolution can be seen in images (a) and (b) of Fig. 2.3. The true flow that represents the Eq.(2.32) is shown in image (c).



(a) (b) (c)

Figure 2.3: Saddle data and true flow – Images (a) and (b) show two later time instances of an initial density that has evolved according to Eq.(2.29) with velocity components given by Eq.(2.32). The vector field in Eq.(2.32) is shown in (c).

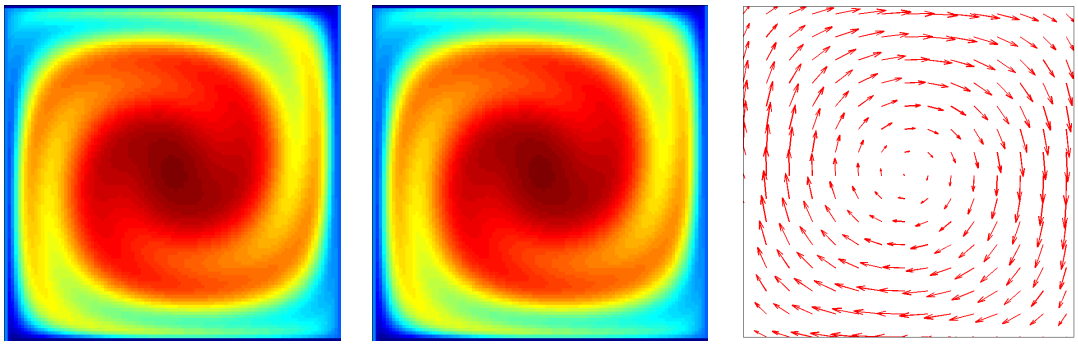
The second data set is a gyre flow (a vortex) represented by the stream function

$$\phi(x, y) = \sin(\pi x) \sin(\pi y) \tag{2.33}$$

on $\mathbb{R}^2$, where we visualize just the subset $[0, 1] \times [0, 1]$. This represents the flow about an elliptic fixed point. We integrate an initial density forward according to Eq.(2.29) with the velocity field given by

$$\langle u, v \rangle = \langle -\pi \sin(\pi x) \cos(\pi y), \pi \cos(\pi x) \sin(\pi y) \rangle. \tag{2.34}$$

After a few iterations, two time instances of the density evolution were selected to check the accuracy of the optical flow algorithms. The two-selected images and the true velocity field that represents in Eq.(2.34) are shown in the images (a), (b) and (c) of Fig. 2.4 respectively.



(a) (b) (c)

Figure 2.4: Gyre data and true flow – Images (a) and (b) show two later time instances of an initial density that has evolved according to Eq.(2.29) with velocity components given by Eq.(2.34) . The true flow field is shown in (c).

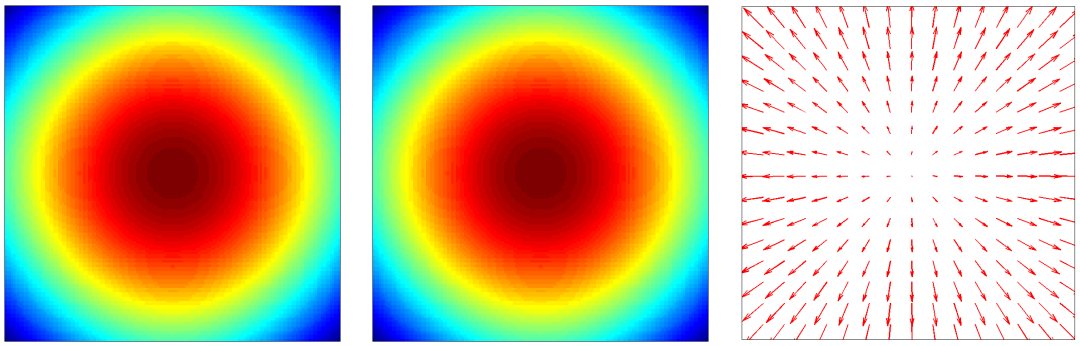
The third example is a flow about a source and unlike the first two examples, this data set is represented by the potential function

$$\psi(x, y) = \sin(x) \cos(y). \tag{2.35}$$

The velocity components are obtained by taking the gradient of the Eq.(2.35) and given by

$$\langle u, v \rangle = \langle \cos x \cos y, -\sin x \sin y \rangle. \tag{2.36}$$

Two time instances of the flow about a source are shown in images (a) and (b) of Fig. 2.5, an evolution given by the velocity field on $[\frac{1}{4}\pi, \frac{3}{4}\pi] \times [\frac{3}{4}\pi, \frac{5}{4}\pi]$. The true flow fields that represent in Eq.(2.35) is shown in image (c).

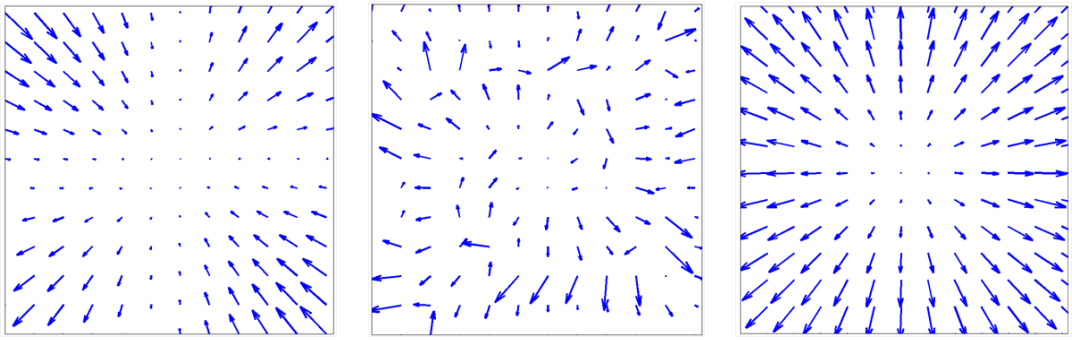


(a) (b) (c)

Figure 2.5: Source data and true flow – Images (a) and (b) show two later time instances of an initial density that has evolved according to Eq.(2.29) with velocity components given by Eq.(2.36). The true flow field which is given in Eq.(2.36) is shown in image (c).

2.4 Results of Optical Flow Calculations

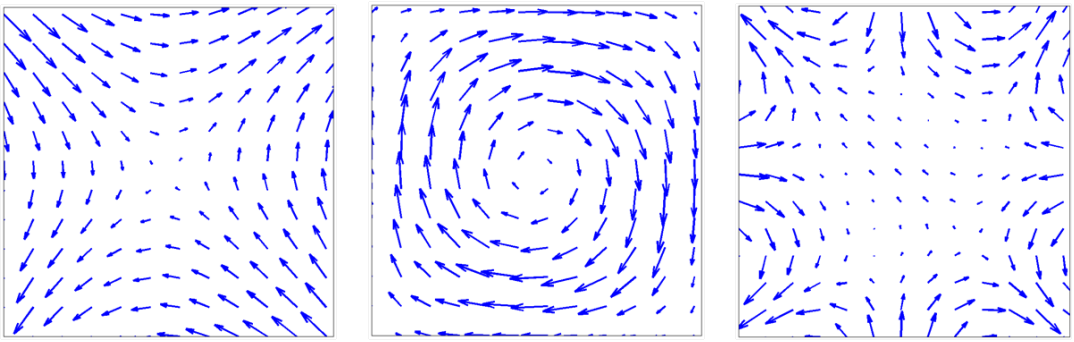
The computed flows from the Gradient Descent formulation are shown in Fig. 2.6. Flows of the saddle, gyre and the source are shown in images (a), (b) and (c) respectively. The gradient descent approach is able to capture the source flow accurately when we compare the reconstructed flow with the true flow. However, the same approach is unable to capture the saddle flow and the gyre flow accurately. This may be due to the fixed step size for each iteration. The determination of step size in each iteration with large number of variables is not an easy task.



(a) (b) (c)

Figure 2.6: Computed flow from gradient descent method – Images (a), (b) and (c) show the computed velocity fields from the Gradient Descent formulation for the saddle, gyre and source images shown in Fig. 2.3, Fig. 2.4 and Fig. 2.5 respectively.

The computed flows from the Gauss-Seidel are shown in Fig. 2.7. Corresponding flows for the saddle, gyre and the source are shown in images (a), (b) and (c) respectively. When we compare the results from the Gauss-Seidel approach with true flow fields, the approach captures the saddle flow and the gyre flow accurately but it does not capture the source flow accurately. If the operator of the system in Eq.(2.26) is diagonally dominant, then the convergence of the Gauss-Seidel method is guaranteed. In these three examples, none of the matrices are diagonally dominant and hence the solution may not converge.



(a) (b) (c)

Figure 2.7: Computed flow from Gauss-Seidel method – Images (a), (b) and (c) show the computed velocity fields from the Gauss-Seidel formulation for the saddle, gyre and source images shown in Fig. 2.3, Fig. 2.4 and Fig. 2.5 respectively.

Since both methods are not capable of capturing all three types of data sets, we have to invent a better way or improve the current method to capture the vector fields of any kind of motion.

Chapter 3

A Stream/Potential Function Method

With every "tick" of the clock in the scientific world, new challenges arise in different domains. Researchers take this opportunity to apply their brand new ideas to the new problems. One such challenge of great interest in the last few decades is analyzing fluid systems using the observed images or a movie of the system. Data is available range from tomography images in medicine through oceanography images to satellite images of the planets. Various applications of analysis of fluid systems range from validating medical image registration in [50], X-ray imaging [51], weather prediction [52], tracking the motion of Arctic sea ice [53] and identifying accurate navigation for planetary landing vehicles [54] demonstrating the importance of our challenge in the literature.

In this Chapter, we introduce a new method to compute optical flow fields, with special emphasis placed on fluid systems. According to The Helmholtz decomposition, a smooth vector field can be decompose into divergence free and curl free components. In this work, we reformulate the optical flow method in terms of a stream function when the flow is divergence free or a potential function when the flow is curl-free

and then derive the velocity components from the computed stream or the potential function.

As we explained in Chapter 2, Horn and Schunck in [9] introduced a formulation for computing optical flow of an object from an observed image sequence. The basic assumption of that formulation is the conservation of the image brightness over time. The brightness constraint, $I_t + I_x u + I_y v = 0$ in Eq.(2.3) can be expressed as an evolution model of

$$I_t(x, y, t) = -\nabla I(x, y, t) \cdot \langle u(x, y, t), v(x, y, t) \rangle, \tag{3.1}$$

which simplifies as

$$I_t = -\left(I_x u + I_y v\right).$$

Our main interest is to compute the velocity field for fluid systems rather than systems with rigid body motion. We do so using measures of image intensity which is the only available source of information. Unlike in rigid body motion, in fluid motion image intensity undergoes distortions in both spatial and temporal directions and hence this is a more challenging problem than problems involving rigid body motion. Most of the work in the optical flow literature assumes that the fluid motion is quasi-rigid and then applies the conservation of image brightness to compute the optical flow, but authors in [29–33] proposed that a fluid system evolves according to the continuity equation and the evolution model can be written as

$$I_t(x, y, t) = -\mathrm{div}\left(I(x, y, t) \langle u(x, y, t), v(x, y, t) \rangle\right). \tag{3.2}$$

Simplifying Eq.(3.2), we can express the evolution model for the continuity equation as

$$I_t = -\left(I_x u + I_y v + I u_x + I v_y\right).$$

The authors introduced the evolution model using continuity equation because the conservation of energy evolution model has some drawbacks in some occasions in fluid motion. One reason is that the image brightness is not conserved when a pixel appears or disappears (occludes) in one image but not in the adjacent image in a time series. Another disadvantage of brightness conservation occurs when the fluid is compressible. In this case, the volume of the fluid changes in the time direction which causes the changes in brightness constancy of the trajectories over the motion. In this chapter, we consider both evolution models for the computation of the optical flow in fluid systems. Recall that the data fidelity corresponding to the conservation of energy is from the Chapter 2 is

$$E(u,v) = \int_{\Omega} \left(I_t + I_x u + I_y v\right)^2 \, d\Omega. \tag{3.3}$$

The data fidelity that represents the continuity equation can be expressed as

$$E(u,v) = \int_{\Omega} \left(I_t + I_x u + I_y v + I u_x + I v_y\right)^2 \, d\Omega. \tag{3.4}$$

Now, we add a suitable regularization term to the data fidelities to make the solutions stable according to the perturbations in the input data. So far we only used the smoothness regularization term

$$R(u,v) = \int_{\Omega} \left(u_x^2 + u_y^2 + v_x^2 + v_y^2\right) \, d\Omega, \tag{3.5}$$

which was introduced by Horn and Schunck. In fluid motions, common features such as gyres, hyperbolic fixed points and sources are not both divergence free and curl free. In most of the places in the fluid either the divergence is high or the curl is high. For this kind of situation, Suter in [34] introduced a "first-order div-curl" regularization term to measure the divergence and curl (vorticity) of the motion separately. This is

done by giving weighting factors a_1 and a_2 on 'div' and 'curl' components separately as

$$R(u,v) = \int_\Omega a_1 \text{div}^2\left(\langle u,v\rangle\right) + a_2 \text{curl}^2\left(\langle u,v\rangle\right)\, d\Omega, \tag{3.6}$$

where $\text{div}\left(\langle u,v\rangle\right) = u_x + v_y$ and $\text{curl}\left(\langle u,v\rangle\right) = v_x - u_y$. The Euler-Lagrange equations corresponding to the "first-order div-curl" regularization term in Eq.(3.6) can be obtained as

$$\begin{aligned} -2a_1 u_{xx} - 2(a_1 - a_2)v_{xy} - 2a_2 u_{yy} &= 0 \\ -2a_2 v_{xx} - 2(a_1 - a_2)u_{xy} - 2a_2 v_{yy} &= 0. \end{aligned}$$

When $a_1 = a_2$, the Euler-Lagrange equations become $-2u_{xx} - 2u_{yy} = 0$ and $-2v_{xx} - 2v_{yy} = 0$ which are the Euler-Lagrange equations for the smoothness regularization introduced by Horn and Schunck in Eq.(3.5). That is, even though the regularization terms in Eq.(3.5) and Eq.(3.6) are different, the minimization problem yields the same solution as the Euler-Lagrange equations are the same. Due to this phenomenon, the authors in [34] proposed a "second-order div-curl" regularization term to incorporate in energy functional to capture the fluid motion from optical flow algorithms. The new regularization term is given as

$$R(u,v) = \int_\Omega \left\|\nabla \text{div}\left(\langle u,v\rangle\right)\right\|^2 + \left\|\nabla \text{curl}\left(\langle u,v\rangle\right)\right\|^2\, d\Omega. \tag{3.7}$$

According to the Helmholtz decomposition, a smooth vector field can be decomposed into divergence free (solenoidal) and curl free (irrotational) components. Further, the two components, divergence free and the curl free, are derived from a stream function ψ and a potential function ϕ respectively. Note that we have to include a laminar flow into the decomposition when non-zero boundary conditions occur. Considering the Helmholtz decomposition and the regularization term in Eq.(3.7), the

authors in [35] developed a new optical flow algorithm to capture the divergence free and curl free components of the flow. In this algorithm, both the stream and potential functions are reconstructed by solving the coupled Euler-Lagrange equations and the flow components are reconstructed along the horizontal and vertical axes. The reconstructed results from this method are promising, but it is hard to implement the algorithm.

This difficulty in implementation and the fact that the algorithm consists of coupled equations motivated us to develop an algorithm [28, 37] that is convenient to implement and easy to solve. In our approach, we concentrate on flows that can be represented by a potential function or a stream function so that the vector field is computed by taking the gradient of the potential function or the symplectic gradient (∇_H) of the stream function. In either case,

$$\begin{aligned} \langle u, v \rangle &= \nabla \phi = \langle \phi_x, \phi_y \rangle \quad \text{or} \\ \langle u, v \rangle &= \nabla_H \psi = \langle -\psi_y, \psi_x \rangle. \end{aligned}$$

The main advantage of this algorithm is that we only need to solve one Euler-Lagrange equation for ϕ or ψ to reconstruct the velocity components u and v. In addition to the simpler computation, the other most important advantage of our algorithm is to impose prior knowledge of the flow on the potential or stream function rather than on the components of the flow. Since it is not always possible to reconstruct a global potential function or a stream function for large scale data sets such as oceanic flows and atmospheric motions, we focus on the local structures such as hyperbolic fixed points, vortices and sources which are readily available in fluid systems.

3.1 A Stream/Potential Function Formulation

First we can rewrite the data fidelities corresponding to both conservation of energy and the continuity equation evolution models in terms of the potential function ϕ and the stream function ψ. The data fidelity in Eq.(3.3) of the potential function formulation is

$$E(\phi) = \int_{\Omega} \left(I_t + I_x\phi_x + I_y\phi_y\right)^2 \, d\Omega, \tag{3.8}$$

and the data fidelity in Eq.(3.4) can be rewritten in terms of the potential function as

$$E(\phi) = \int_{\Omega} \left(I_t + I_x\phi_x + I_y\phi_y + I\phi_{xx} + I\phi_{yy}\right)^2 \, d\Omega. \tag{3.9}$$

Similar to the potential function formulation, we can obtain the stream function formulation for the data fidelity in Eq.(3.3) as

$$E(\psi) = \int_{\Omega} \left(I_t - I_x\psi_y + I_y\psi_x\right)^2 \, d\Omega, \tag{3.10}$$

and the data fidelity in Eq.(3.4) as

$$E(\psi) = \int_{\Omega} \left(I_t - I_x\psi_y + I_y\psi_x - I\psi_{yx} + I\psi_{xy}\right)^2 \, d\Omega. \tag{3.11}$$

When we minimize the optical flow energy functional, the algorithms for the potential function and the stream function are similar to each other. Therefore, for simplicity, we present only the stream function formulation in the rest of the chapter. Also, for the sake simplification, we write both data fidelities in Eq.(3.10) and Eq.(3.11) in terms of two operator matrices A_{CI} and A_{CE}, where A_{CI} is the operator corresponding to the Conservation of Intensity (CI) model and A_{CE} corresponds to the Continuity

Equation (CE) model. The two data fidelities in the new form are

$$E_{CI}(\psi) = \int_{\Omega} (I_t + A_{CI}\psi)^2 \, d\Omega, \quad \text{where} \quad A_{CI} = -I_x D_y + I_y D_x \quad \text{and} \tag{3.12}$$

$$E_{CE}(\psi) = \int_{\Omega} (I_t + A_{CE}\psi)^2 \, d\Omega,$$

$$\text{where} \quad A_{CE} = -I_x D_y + I_y D_x - I D_{yx} + I D_{xy}. \tag{3.13}$$

Here the operators $D_{\bullet\bullet}$ are arrays of size $m \times m$ to compute the partial derivatives of a given vector of size $m \times 1$ with respect to indices $\bullet\bullet$. Here we stack the given array of size $p \times q$ into $k \times 1$ vector and $k = pq$. To develop the operators $D_{\bullet\bullet}$, we use finite difference approximations with suitable boundary conditions. Next, we will illustrate a construction of such a derivative operator matrix.

3.1.1 Construction of a Derivative Operator

Suppose we have a function $M(x, y) \in \mathbb{R}^{p \times q}$ which is a $2D$ array of real numbers with p rows and q columns. Now we have to compute the partial derivative of M with respect to x. First we construct a derivative operator D_x of size $k \times k$, where $k = pq$ such that we can compute the partial derivative of M with respect to x. To illustrate the constructions of the derivative operator, we consider a fourth-order finite difference approximation to compute the pointwise partial derivative of the array M with respect to x. For a point $m_{ij} = M\left(x\left(i,j\right), y\left(i,j\right)\right)$ of the array, the finite difference approximation is

$$\frac{\partial m_{ij}}{\partial x} = \frac{1}{12h}\left[m_{i,j-2} - 8m_{i,j-1} + 0m_{i,j} + 8m_{i,j+1} - m_{i,j+2}\right], \tag{3.14}$$

where h is the width of the uniform grid in both the x and y directions, and i and j are the row and column numbers respectively.

We now consider an example of an array $M(x, y)$ of size 3×5 where we want to compute $M_x(x, y)$. We index the elements in the array in the form of a column vector as shown in Table 3.1 and the elements in $M(x, y)$ are defined as $m_i = M(x(i), y(i))$ for $i = 1, 2, ..., 15$.

Table 3.1: The table shows the elements of a given $2D$ array of size 3×5 and they are labeled by considering the given array as a column vector.

m_1	m_4	m_7	m_{10}	m_{13}
m_2	m_5	m_8	m_{11}	m_{14}
m_3	m_6	m_9	m_{12}	m_{15}

When we compute the partial derivatives, we must define suitable boundary conditions. Without loss of generality we choose reflexive boundary conditions for this example. After including the boundary points for the array M, the resulting array is shown in Table 3.2. In Table 3.2, we add two new boundary columns to both the left and the right sides of the array. These new columns are shown in bold letters and $\boldsymbol{m_i} = m_i$ for $i = 1, 2, ..., 15$. The number of boundary points that are added depends on the size of the finite difference stencil.

Table 3.2: The table shows the elements of a given array of size 3×5 with added boundary points. We assume that the reflexive boundary conditions are appropriate and added two columns to the both left and right of the table. The boundary points are highlighted in bold letters.

$\boldsymbol{m_4}$	$\boldsymbol{m_1}$	m_1	m_4	m_7	m_{10}	m_{13}	$\boldsymbol{m_{13}}$	$\boldsymbol{m_{10}}$
$\boldsymbol{m_5}$	$\boldsymbol{m_2}$	m_2	m_5	m_8	m_{11}	m_{14}	$\boldsymbol{m_{14}}$	$\boldsymbol{m_{12}}$
$\boldsymbol{m_6}$	$\boldsymbol{m_3}$	m_3	m_6	m_9	m_{12}	m_{15}	$\boldsymbol{m_{15}}$	$\boldsymbol{m_{13}}$

Next we compute the partial derivative of each element of M with respect to

x using the approximation in Eq.(3.14) as explained. If we compute the partial derivative of m_1 with respect to x, the resulting approximation is obtained as

$$\begin{aligned}\frac{\partial m_1}{\partial x} &= \frac{1}{12h}\left[\boldsymbol{m_4} - 8\boldsymbol{m_1} + 0m_1 + 8m_4 - m_7\right] \\ &= \frac{1}{12h}\left[-8m_1 + 9m_4 - m_7\right].\end{aligned}$$

The approximations for the partial derivatives of m_2 and m_3 with respect to x also follow the same pattern. Further, we can estimate the partial derivative of m_5 with respect to x using the approximation in Eq.(3.14) as

$$\begin{aligned}\frac{\partial m_5}{\partial x} &= \frac{1}{12h}\left[\boldsymbol{m_2} - 8m_2 + 0m_5 + 8m_8 - m_{11}\right] \\ &= \frac{1}{12h}\left[-7m_2 + 8m_8 - m_{11}\right].\end{aligned}$$

Continuing in a similar manner for all 15 elements, we can construct the the derivative operator matrix D_x for any real matrix of size $p \times q$ as the product of $\frac{1}{12h}$ and the Table 3.3. The operator $12hD_x$ is given in the following table.

Table 3.3 shows the elements of a derivative operator to compute the partial derivative of a given real valued 3×5 array. In this case, we compute the derivatives using finite difference approximations as explained in Eq.(3.14) with reflexive boundary conditions, and there is a multiplicative factor of $12h$ on each element on the operator matrix. The resulting matrix is sparse as it has only five non-zero diagonals. We can extend the construction for any number of rows and columns as well as partial derivatives with respect to other variable or combination of variables.

Table 3.3: The table shows the operator matrix needed to compute partial derivative of any real valued array of size 3×5 with respect to x. All the entries have a multiplication factor of $\frac{1}{12h}$ and we must include this when we compute the derivatives. For the computation we use the fourth order finite difference approximation, as shown in Eq.(3.14) with reflexive boundary conditions. The resulting matrix has five non-zero diagonals.

	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}	m_{13}	m_{14}	m_{15}
m_1	**-8**	0	0	**9**	0	0	**-1**	0	0	0	0	0	0	0	0
m_2	0	**-8**	0	0	**9**	0	0	**-1**	0	0	0	0	0	0	0
m_3	0	0	**-8**	0	0	**9**	0	0	**-1**	0	0	0	0	0	0
m_4	**-7**	0	0	**0**	0	0	**8**	0	0	**-1**	0	0	0	0	0
m_5	0	**-7**	0	0	**0**	0	0	**8**	0	0	**-1**	0	0	0	0
m_6	0	0	**-7**	0	0	**0**	0	0	**8**	0	0	**-1**	0	0	0
m_7	**1**	0	0	**-8**	0	0	**0**	0	0	**8**	0	0	**-1**	0	0
m_8	0	**1**	0	0	**-8**	0	0	**0**	0	0	**8**	0	0	**-1**	0
m_9	0	0	**1**	0	0	**-8**	0	0	**0**	0	0	**8**	0	0	**-1**
m_{10}	0	0	0	**1**	0	0	**-8**	0	0	**0**	0	0	**7**	0	0
m_{11}	0	0	0	0	**1**	0	0	**-8**	0	0	**0**	0	0	**7**	0
m_{12}	0	0	0	0	0	**1**	0	0	**-8**	0	0	**0**	0	0	**7**
m_{13}	0	0	0	0	0	0	**1**	0	0	**-9**	0	0	**8**	0	0
m_{14}	0	0	0	0	0	0	0	**1**	0	0	**-9**	0	0	**8**	0
m_{15}	0	0	0	0	0	0	0	0	**1**	0	0	**-9**	0	0	**8**

Next we will present visualization of a derivative operator under two different boundary conditions. Fig. 3.1 shows the partial derivative operator D_y under two different boundary conditions. The operator is designed to compute the partial derivative of a given 6×5 array with respect to y. The operator in image (a) is constructed using reflexive boundary conditions, and the operator in image (b) is constructed using zero boundary conditions. These matrices are sparse, and in the reflexive boundary case, the number of non-zero elements are 100 among 900 while that of zero boundary conditions is 90. In terms of these operators, we can rewrite the functionals in a simple way.

In the optical flow algorithm, the total energy functional is a combination of the data fidelity and the regularization term with a weighting factor α. In the minimization of the total energy functional, we use a variational approach to reach the minimum solution. For this purpose, we need the gradient (first variation) of the

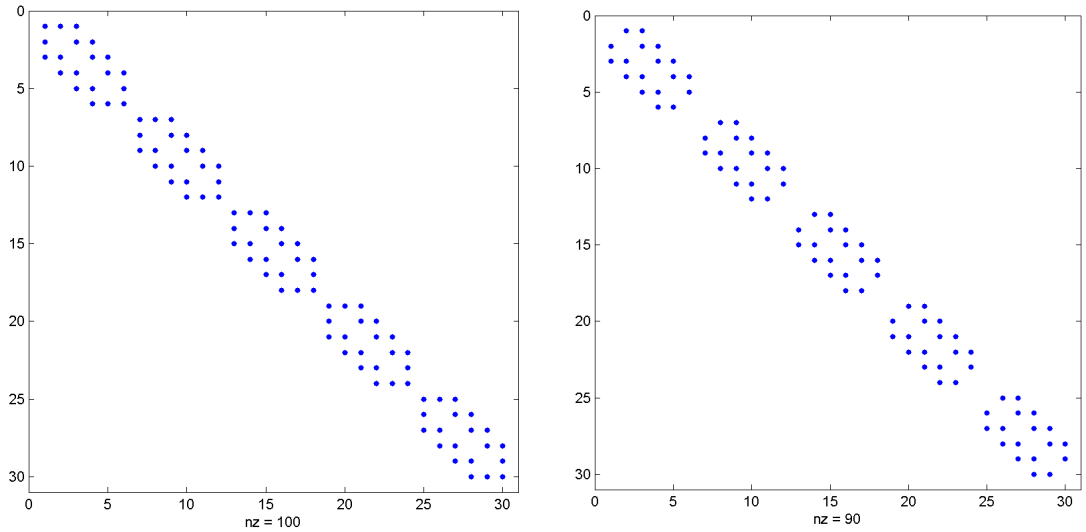


Figure 3.1: D_y with two different boundary conditions – Image (a) and (b) show two views of operator matrices to compute the partial derivative of a given 6×5 matrix with respect to y. The operator D_y in image (a) uses reflexive boundary condition and in that case the number of non-zero elements is 100 out of 900. However, the D_y operator in image (b) uses zero boundary condition and then the number of non-zero elements is 90 out of 900.

energy functionals. We can determine the gradient of the data fidelities in Eq.(3.12) and Eq.(3.13) as

$$\nabla E_{CI} = 2A_{CI}^{*}(I_t + A_{CI}\psi) \qquad \text{and} \qquad \nabla E_{CE} = 2A_{CE}^{*}(I_t + A_{CE}\psi),$$

respectively. Here A_{CI}^{*} is the adjoint operator of A_{CI} and vice versa. Note that the Hessian (second variation) operators for the data fidelities in Eq.(3.12) and Eq.(3.13) are

$$\nabla^2 E_{CI} = 2A_{CI}^{*}A_{CI} \qquad \text{and} \qquad \nabla^2 E_{CE} = 2A_{CE}^{*}A_{CE}.$$

The Hessian matrices are positive-semi definite; hence both data fidelities are convex functionals. The optical flow energy functional consists of the data fidelity and the regularization term. Therefore, with an appropriate regularization term $R(\psi)$ and a

regularization parameter α, the total energy functionals for the conservation of energy model and the continuity equation can be written as

$$E(\psi) = \int_\Omega \left(I_t + A_{CI}\psi\right)^2 d\Omega + R(\psi) \quad \text{and} \tag{3.15}$$

$$E(\psi) = \int_\Omega \left(I_t + A_{CE}\psi\right)^2 d\Omega + R(\psi). \tag{3.16}$$

One of the simple regularizations which provides the convexity to the energy functional is

$$R_1(\psi) = \int_\Omega \psi^2 + \psi_x^2 + \psi_y^2 + \psi_{xx}^2 + \psi_{yy}^2 \, d\Omega. \tag{3.17}$$

Since there does not exist a u-v formulation for this regularization term, we are not interested in using this regularization in this chapter, but we will demonstrate it in Chapter 4. Two other important regularization terms are the smoothness regularization term of which the u-v formulation is in Eq.(3.5), denoted by $R_2(\psi)$

$$R_2\left(\psi\right) = \int_\Omega \left(\psi_{xx}^2 + \psi_{yy}^2 + \psi_{xy}^2 + \psi_{yx}^2\right) d\Omega, \tag{3.18}$$

with the gradient

$$GR_2(\psi) = (B_2 + B_2^*)\psi, \quad \text{where} \quad B_2 = D_{xx}^* D_{xx} + D_{yy}^* D_{yy} + D_{xy}^* D_{xy} + D_{yx}^* D_{yx}$$

and the regularization term $R_3(\psi)$, which regularizes u and v components is

$$R_3\left(\psi\right) = \int_\Omega \left(\psi_x^2 + \psi_y^2\right) d\Omega, \tag{3.19}$$

with the gradient of $GR_3(\psi) = (B_3 + B_3^*)\psi$, where $B_3 = D_x^* D_x + D_y^* D_y$. Note that the u-v formulation of the regularization term can be written as

$$R_3\left(u, v\right) = \int_\Omega \left(u^2 + v^2\right) d\Omega. \tag{3.20}$$

The stream function formulation is valid for divergence free flows only. On the other hand, if the flow is curl free, the potential function formulation can be applied to determine velocity fields. The two data terms of conservation of intensity and the continuity equation are

$$E_{CI}(\phi) = \int_\Omega \left(I_t + A_{CI}\phi\right)^2 d\Omega, \quad \text{where} \quad A_{CI} = I_x D_x + I_y D_y \quad \text{and} \tag{3.21}$$

$$E_{CE}(\phi) = \int_\Omega \left(I_t + A_{CE}\phi\right)^2 d\Omega,$$

$$\text{where} \quad A_{CE} = I_x D_x + I_y D_y + I D_{xx} + I D_{yy}. \tag{3.22}$$

Now the regularization terms $R_2(\psi)$ and $R_3(\psi)$ can be written in the form of potential function as

$$R_2(\phi) = \int_\Omega \left(\phi_{xx}^2 + \phi_{yy}^2 + \phi_{xy}^2 + \phi_{yx}^2\right) d\Omega \quad \text{and} \tag{3.23}$$

$$R_3(\phi) = \int_\Omega \left(\phi_x^2 + \phi_y^2\right) d\Omega. \tag{3.24}$$

The gradients are similar to the stream function formulation for both regularization terms and as we mentioned earlier, the numerical approach for solving the potential function problem is similar to the stream function problem. According to the way we developed the data fidelities and the regularization terms, we have two different approaches, the u-v method and the stream function method. We develop different algorithms by combining two data fidelities and two regularization terms. When we minimize all these algorithms, we use a variational approach. In the stream function approach, we set up the problem by setting the Euler-Lagrange equation to be zero. Both the data terms and the regularization terms give a linear gradient in all formulations. Hence the Euler-Lagrange equations for all the above methods are linear. For a simpler presentation, we will discuss the algorithm for the stream func-

tion formulation with the conservation of intensity data fidelity and the smoothness regularization term. The corresponding Euler-Lagrange equation is given by

$$\left[2A_{CI}^{*}A_{CI} + \alpha\left(B_2 + B_2^{*}\right)\right]\psi = -2A_{CI}^{*}I_t. \tag{3.25}$$

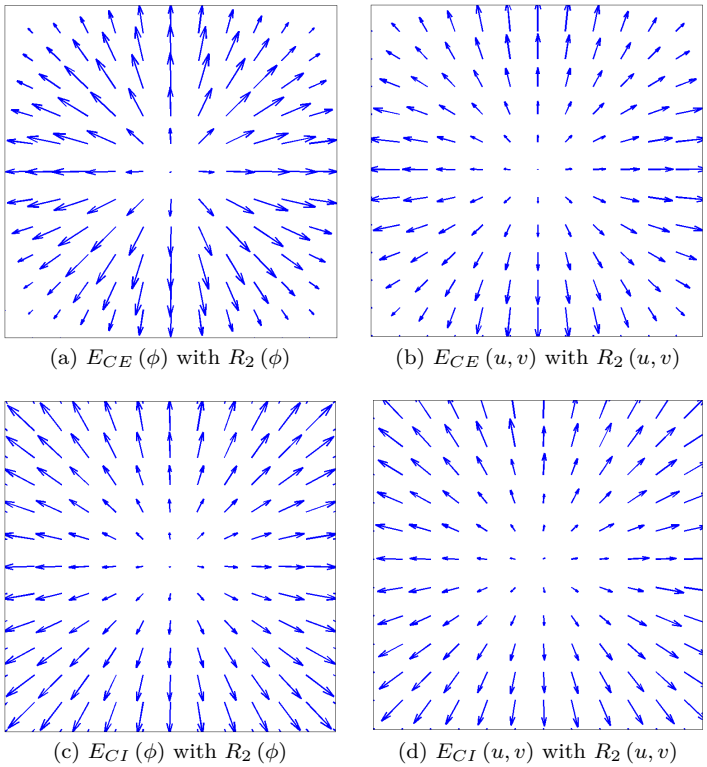
Either an iterative method or direct solution method for solving sparse linear systems can be applied to the system in Eq.(3.25) to determine the optimal solution ψ^*. As a direct way of reaching the solution, we use LU factorization on $2A_{CI}^{*}A_{CI} + \alpha\left(B_2 + B_2^{*}\right)$ and then use Gaussian elimination to determine the solution. Most of the time, the Euler-Lagrange equations corresponding to the regularization terms are linear and symmetric. In the symmetric case $B^* = B$, as B is real-valued matrix leading to the system $\left(A_{CI}^{*}A_{CI} + \alpha B_2^{*}\right)\psi = -A^{*}I_t$. To solve systems of this form, we can apply the Cholesky factorization. In [55] however, we see some instances where the regularization terms such as the total variation provides non-linearity in the Euler-Lagrange equation. In that case, we cannot apply the direct linear methods to solve the system and instead apply a suitable iterative method.

3.2 Results from the Synthetic Flows

We have two data fidelities corresponding to the conservation of intensity and the continuity equation. Combining these two with the two regularization terms R_2 and R_3, in each stream function and u-v formulations, eight different algorithms are obtained. We can now apply these algorithms on synthetic data sets which represent often visible local structures in fluids such as a hyperbolic fixed point, a vortex and a source. We constructed these data sets by evolving an initial density according to the known velocity fields and then selected two time instances of the density evolution as explained in Sec. 2.3. Note that in this Chapter we discuss only the qualitative difference of the stream function formulation and the u-v formulations using the two

data fidelities and the two regularization terms on the synthetic data sets. Further in Chapter 4 we present a quantitative analysis of the stream function formulation and the u-v on the same data sets using the two data fidelities and six different regularization terms including R_2 and R_3 in terms of the Mean Angular Errors.

The first example is the source flow as shown in Fig. 2.5. Recall that the true velocity components are given by $\langle u, v\rangle = \langle \cos x \cos y, -\sin x \sin y)\rangle$ and hence the $\text{div}\left(\langle u, v\rangle\right) = -\sin x \cos y$, which is not equal to zero in the complete domain. Therefore, the flow is curl free and it can be represented by a potential function. Hence, the potential function formulation is appropriate for this data set.

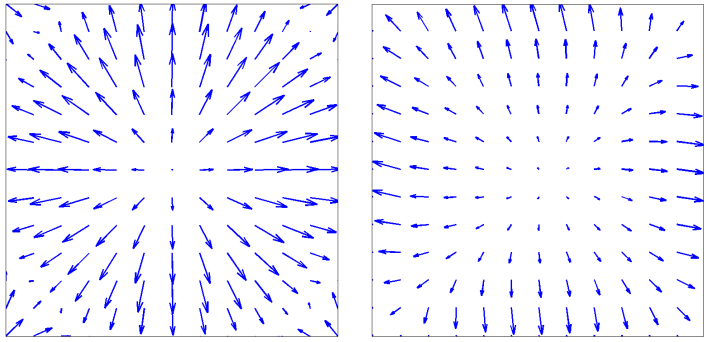


(a) $E_{CE}(\phi)$ with $R_2(\phi)$ (b) $E_{CE}(u, v)$ with $R_2(u, v)$

(c) $E_{CI}(\phi)$ with $R_2(\phi)$ (d) $E_{CI}(u, v)$ with $R_2(u, v)$

Figure 3.2: Source with R_2 – Reconstructed flow fields for source data from the potential function formulation shown in first column and u-v formulation in second column using the regularization term R_2. Flow fields in the first row are computed using continuity equation data fidelity whereas the second row using the conservation of intensity data fidelity. Results from both formulations with R_2 capture the source flow field accurately.

Images in Fig. 3.2 represent the reconstructed vector fields for the source data from the potential function formulation and the u-v formulation with the regularization term R_2. Images in the first row are reconstructed using the continuity equation data fidelity, whereas the images in the second row are computed using the conservation of intensity data fidelity. Also the first column represents the reconstructions from the stream function formulation while the second column shows the reconstructions from the u-v formulation. All four reconstructions are reasonable, but near the corners, the formulations with the continuity equation data fidelity have some difficulties capturing the correct magnitudes.

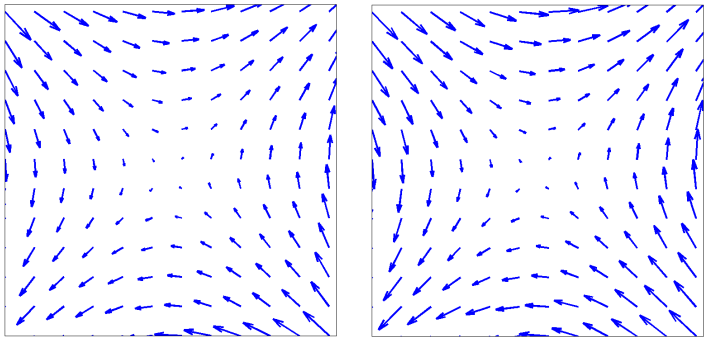


(a) $E_{CE}(\phi)$ with $R_3(\phi)$ (b) $E_{CE}(u,v)$ with $R_3(u,v)$

(c) $E_{CI}(\phi)$ with $R_3(\phi)$ (d) $E_{CI}(u,v)$ with $R_3(u,v)$

Figure 3.3: Source with R_3 – Computed flow fields for the source data from the potential function formulation (first column) and u-v formulation (second column) using the regularization term R_3. Flow fields in the first row are computed using continuity equation data fidelity whereas the second row using the conservation of intensity data fidelity. Results from both frameworks are reasonable other than near the boundaries of the flow fields from continuity equation data fidelity.

Fig. 3.3 shows the reconstructed flow fields for the potential function formulation and the u-v formulation with R_3 regularization term on the source data. The reconstructions from the continuity equation data fidelity are shown in the first row and the conservation of intensity data fidelity in the second row. Flows from the the potential function approach and the u-v approach are displayed in the first and second columns respectively. Reconstructions from the conservation of image intensity data fidelity captures the source flow accurately; however, the continuity equation data fidelity with both frameworks show larger errors near the boundaries. Since the two data fidelities have unequal gradients, we may expect these kind of differences.



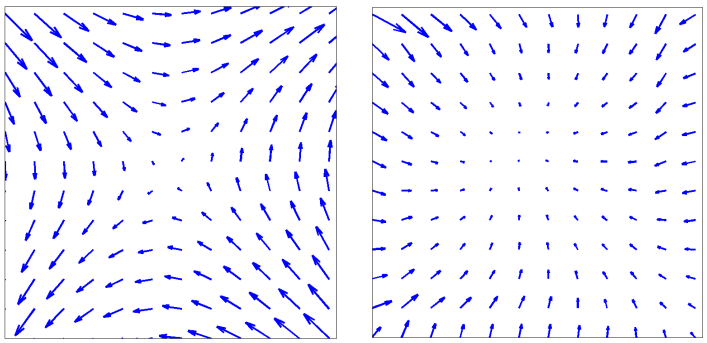
(a) $E_{CE}(\psi)$ with $R_2(\psi)$ (b) $E_{CE}(u,v)$ with $R_2(u,v)$

(c) $E_{CI}(\psi)$ with $R_2(\psi)$ (d) $E_{CI}(u,v)$ with $R_2(u,v)$

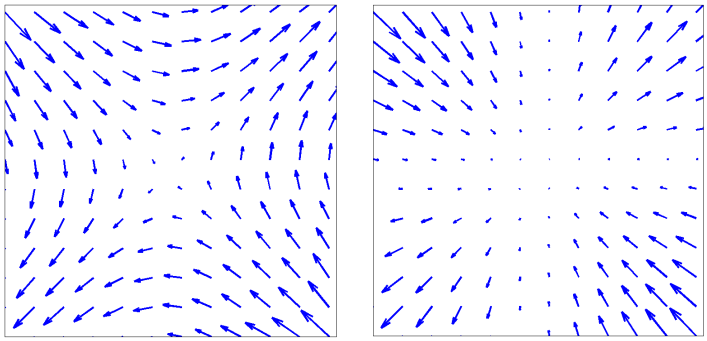
Figure 3.4: Hyperbolic Flow with R_2 – Reconstructed flow fields for the Hyperbolic data from the stream function formulation (first column) and u-v formulation (second column) using the regularization term R_2. Flow fields in the first row are computed using continuity equation data fidelity whereas the second row flow fields are from conservation of intensity data fidelity. Both stream function and the u-v formulations with R_2 reconstruct the hyperbolic fixed point accurately.

The second instance where we apply the eight different algorithms is on a hyperbolic fixed point. The true flow field corresponding to this data set is $\langle u, v\rangle = \langle 2y, 2x\rangle$. The divergence of the flow is $\operatorname{div}(\langle u, v\rangle) = 0$; hence we apply the stream function formulation on the hyperbolic data set. First, we apply both the stream function approach and the u-v approach with the regularization term R_2. In this case, all the reconstructions capture the hyperbolic fixed point accurately. In Fig. 3.4, the first row presents the result from the continuity equation data fidelity and the image represents the flow from the stream function formulation whereas the second image presents the result from the u-v formulation. Similarly, the second row represents the reconstructions from the the conservation of intensity data fidelity where the left image is from the stream function formulation while the right image is from the u-v formulation.

Now we apply both the stream function approach and the u-v approach with the regularization term R_3 on the hyperbolic fixed point. In this case, all the reconstructions from the stream function formulation capture the hyperbolic fixed point successfully whereas the u-v formulation does not reconstruct the flow accurately. The results are shown in Fig. 3.5 where the flow field representation in the figure is similar to the R_2 case in Fig. 3.4. These results outperform the stream function formulation over the u-v formulation. Further, the results indicate that the optimal flow fields from regularization term R_3 under both data fidelities does not produce a hyperbolic fixed point in u-v formulation. In other words, the u-v approach with regularization term R_3 is unable to capture some stream flows under either of the data fidelities.



(a) $E_{CE}(\psi)$ with $R_3(\psi)$ (b) $E_{CE}(u,v)$ with $R_3(u,v)$

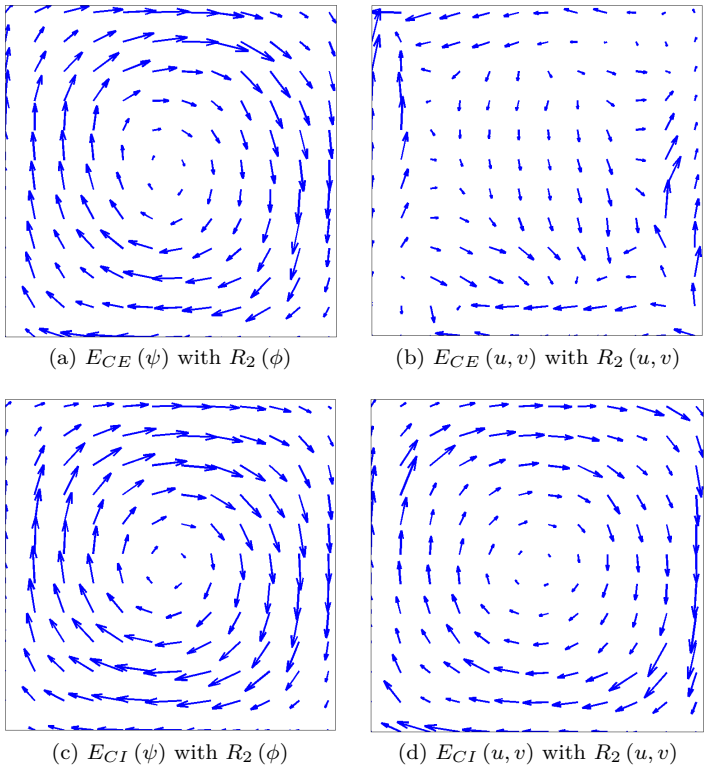


(c) $E_{CI}(\psi)$ with $R_3(\psi)$ (d) $E_{CI}(u,v)$ with $R_3(u,v)$

Figure 3.5: Hyperbolic Flow with R_3 – Reconstructed flow fields for the hyperbolic data from the stream function formulation (first column) and u-v formulation (second column) using the regularization term R_2. Flow fields in the first row are computed using continuity equation data fidelity whereas the second row flow fields are from conservation of intensity data fidelity. Reconstructed flow fields from the stream function method with both data fidelities are successful, but the u-v approach with both data fidelities is unable to capture the hyperbolic fixed point.

The last synthetic local structure that we apply the eight different algorithms to is a data set which represents a vortex (gyre) as we showed in Fig. 2.4. The true velocity field that represents the data set is $\langle u, v\rangle = \langle -\pi \sin(\pi x)\cos(\pi y), \pi \cos(\pi x)\sin(\pi y)\rangle$. The divergence of the flow, $\mathrm{div}(\langle u, v\rangle) = 0$, and hence flow is incompressible, so we apply the stream function formulation on the gyre data set to compare with the u-v formulation.

Fig. 3.6 shows the reconstructed flow fields for the gyre data set using the regularization term R_2. The stream function method reconstructs the gyre flow successfully with both data fidelities whereas the u-v formulation captures the gyre flow only with

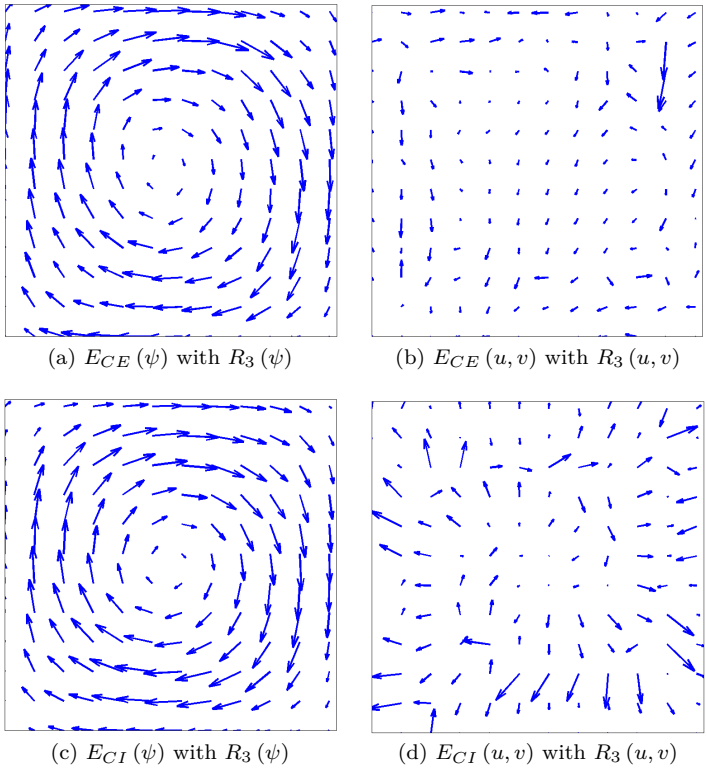


(a) $E_{CE}(\psi)$ with $R_2(\phi)$ (b) $E_{CE}(u,v)$ with $R_2(u,v)$

(c) $E_{CI}(\psi)$ with $R_2(\phi)$ (d) $E_{CI}(u,v)$ with $R_2(u,v)$

Figure 3.6: Gyre Flow with R_2 – Reconstructed flow fields for the gyre data from the stream function formulation (first column) and u-v formulation (second column) using the regularization term R_2. Flow fields in the first row are computed using continuity equation data fidelity whereas the second row flow fields are from conservation of intensity data fidelity. Both flow fields from the stream function formulation with both data fidelities are successfully reconstructed. In the u-v formulation, only the data fidelity from the conservation of intensity captures the flow successfully, whereas continuity equation data fidelity does not capture the gyre flow.

the conservation of intensity data fidelity which is shown in the lower right image.



(a) $E_{CE}(\psi)$ with $R_3(\psi)$ (b) $E_{CE}(u,v)$ with $R_3(u,v)$

(c) $E_{CI}(\psi)$ with $R_3(\psi)$ (d) $E_{CI}(u,v)$ with $R_3(u,v)$

Figure 3.7: Gyre Flow with R_3 – Reconstructed flow fields for the gyre data from the stream function formulation (first column) and u-v formulation (second column) using the regularization term R_3. Flow fields in the first row are computed using continuity equation data fidelity whereas the second row flow fields are from conservation of intensity data fidelity. Only stream function formulation with both data fidelities successfully reconstructed gyre flow, whereas none of the data fidelity with u-v approach captured the gyre flow.

As shown in Fig. 3.7, these reconstructions lead to a positive conclusion regarding the stream function method. In Fig. 3.7, the u-v formulation with both continuity equation data fidelity (upper-right) and the conservation of intensity data fidelity (lower-right) are not capable of reconstructing the gyre flow. The stream function reconstruction methods, however, reconstruct the gyre flow quite accurately. This is another example where the stream function formulation outperforms the u-v formulation.

3.3 Application and Examples

Since we have developed the algorithm for capturing fluid motions for real world applications, we need to check the accuracy of our method with real data. Therefore, we applied the stream function optical flow formulation to capture flows in two oceanographic data sets. These oceanic flows are three-dimensional and the affected by the coriolis force balanced by pressure gradients. A two-dimensional assumption is often used in scenarios where the flow is primarily on the surface observed or if there is an inherent 2D symmetry. For example, in plankton dynamics, the evolution often tends to be in the first few centimeters.

3.3.1 An Example Using Sea Surface Temperature Data

In this illustration, we consider an image sequence generated from a Regional Ocean Model System (ROMS) which represents the Sea Surface Temperature (SST) off the coast of Oregon, U.S.A [56]. This data was provided by Nicholas B. Tufillaro, John Osborne and Alex Kurapov from their Regional Ocean Modeling System (ROMS) in August 2002. We thank Nicholas B. Tufillaro, John Osborne and Alex Kurapov for providing data. This complete sequence of images was taken in the first week of August in 2002. The first image of this sequence is shown in Fig. 3.8. Orange represents high temperature, red represents low temperature and green represents land.

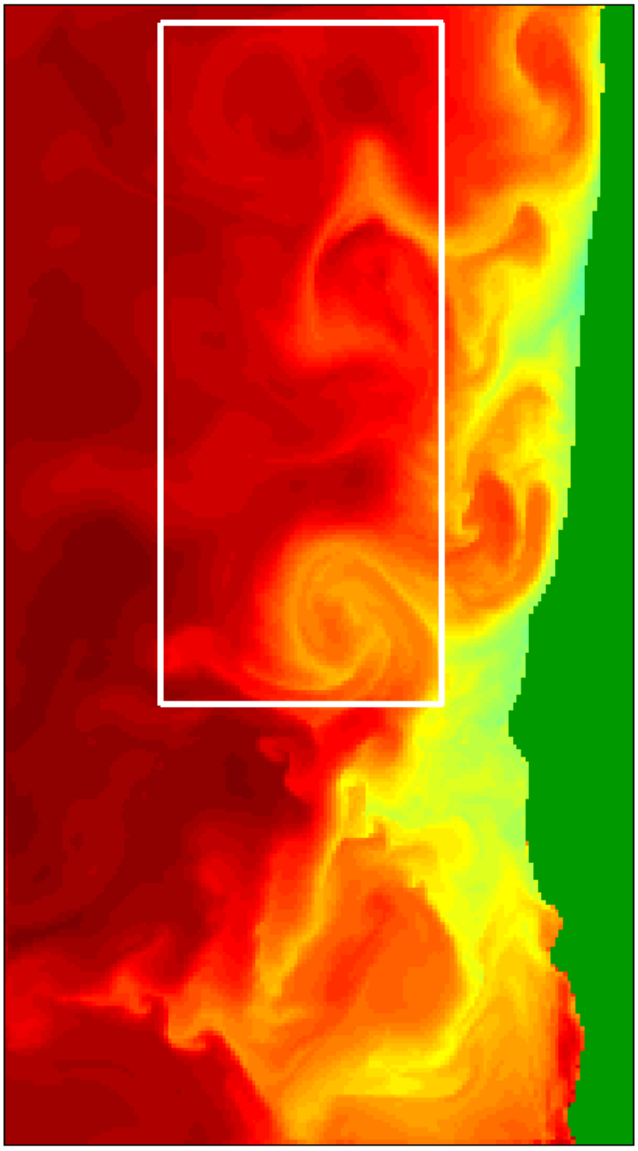


Figure 3.8: SST full image – Image shows the variations of the sea surface temperature off the coast of Oregon, USA. Since the local structures are not clearly visible, a selected area from the white rectangle is used for the calculations.

The image in Fig. 3.8 represents the complete area we are interested in which is 540 by 300 km with 1 km horizontal resolution. The approximate location can be expressed as North latitude from 41 to 46 degrees and West longitude from -125 to -124. Withing this image the highlighted rectangular area features a local structure which represents three individual gyres. The images (a) and (b) in the Fig. 3.9 show two time instances of an image sequence taken one hour apart. The data set is gen-

erated from an 3D ocean model along the Oregon shelf that consists of wind-driven and tidal flows evolving some initial satellite data [57]. The model is verified against the sea surface temperature data from the Geostationary Operational Environmental Satellite (GOES).

For the oceanic data, when the flow is divergence free, the stream function formulation is appropriate and the flow is curl-free, the potential function method is appropriate. To highlight the performance of the stream function method in nearly divergence free flows, we compute the solution from stream function method, the u-v formulation as well as the potential function formulation on the SST data. This leads to a comparison of stream function vs potential function vs u-v formulation. In our results for the SST data, the continuity equation data fidelity was unable capture the flow field with each of these algorithms with R_2 and R_3. Therefore, we present the results only from the conservation of intensity data fidelity with the regularization term R_2 and R_3. In Fig. 3.9, images (c) and (d) show the reconstructed flow fields from the stream function formulation R_2 and R_3 respectively. The images (e) and (f) represents the results from the potential function formulation with R_2 and R_3 respectively. Reconstructed flow fields from the u-v formulation with the regularization terms R_2 and R_3 are shown in images (g) and (h) respectively. The stream function formulation with regularization term R_2 captures the gyres (vortices) accurately, but R_3 does not fully capture the gyres. None of the potential functions or u-v formulations, however, were able to capture any of the structure from the flow. According to these results, it can be concluded that in order to analyze the dynamics of the flow, it is best to assume the incompressibility of the flow and to compute the vector field using the stream function formulation. This vector field can then be used to determine things like mass transport and coherent sets. Beginning directly from the image data allows us to determine the flow dynamics without the use of the model [58].

3.3.2 Example from GOCI satellite

Our second example is also an oceanic fluid system. In this example, first we compute the velocity field between two given images and then to verify the accuracy of the stream function method, we reconstruct the second image by integrating the first image using the computed velocity field. The data set represents product movements in the seas of South Korea which was obtained from the on-board optical sensor Geostationary Ocean Color Imager (GOCI) [59] of the Communication, Ocean, and Meteorological Satellite (COMS) from South Korea. The temporal resolution between images (a) and (b) in Fig. 3.10 is 10 minutes. We applied both the stream function and the potential function formulations with regularization R_2 and the stream function formulation gave a reasonable solution which is shown in image (c) in Fig. 3.10. In the absence of a known vector field, a study of image evolution can serve to validate the method. Next we reconstruct the second image (b) in Fig. 3.10 by integrating the first image (a) in Fig. 3.10 forward in time under the evolution model (3.2) with the computed flow field shown in image (c) in Fig. 3.10. The reconstructed second image is shown in image (d) in Fig. 3.10.

To verify the accuracy of the reconstructed image, we compare the relative error between the actual image and the reconstructed image. The relative error was computed over the image domain and then a single number was obtained to compare the errors. We compute the average of the magnitude of the error over the domain, called the mean relative error. We apply both the stream function formulation and the u-v formulation on images (a) and (b) in Fig. 3.10 for the comparison. The mean relative error for the stream function approach is 2.21 % and for the u-v approach is 5.32 % validating the accuracy of the stream function formulation over the u-v approach.

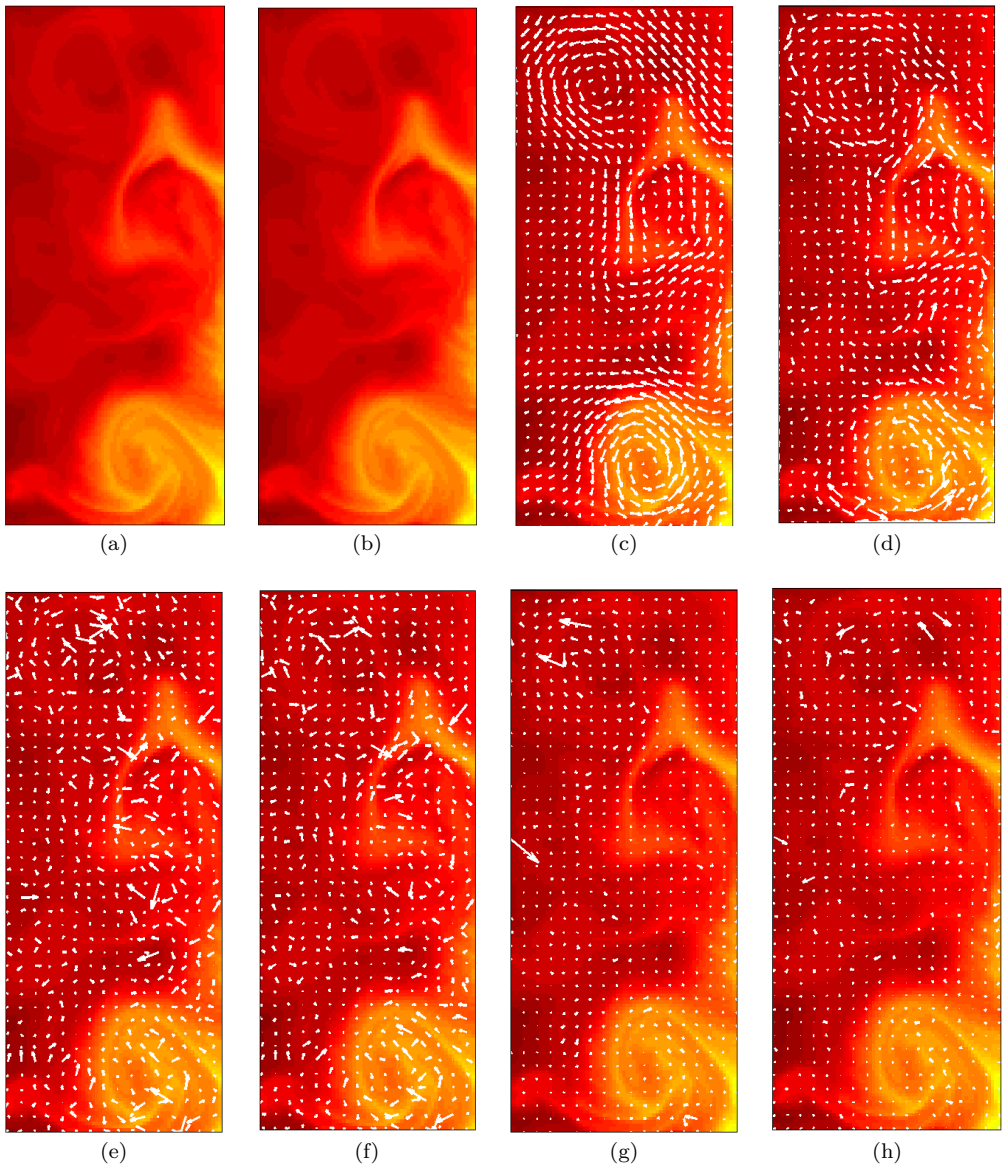
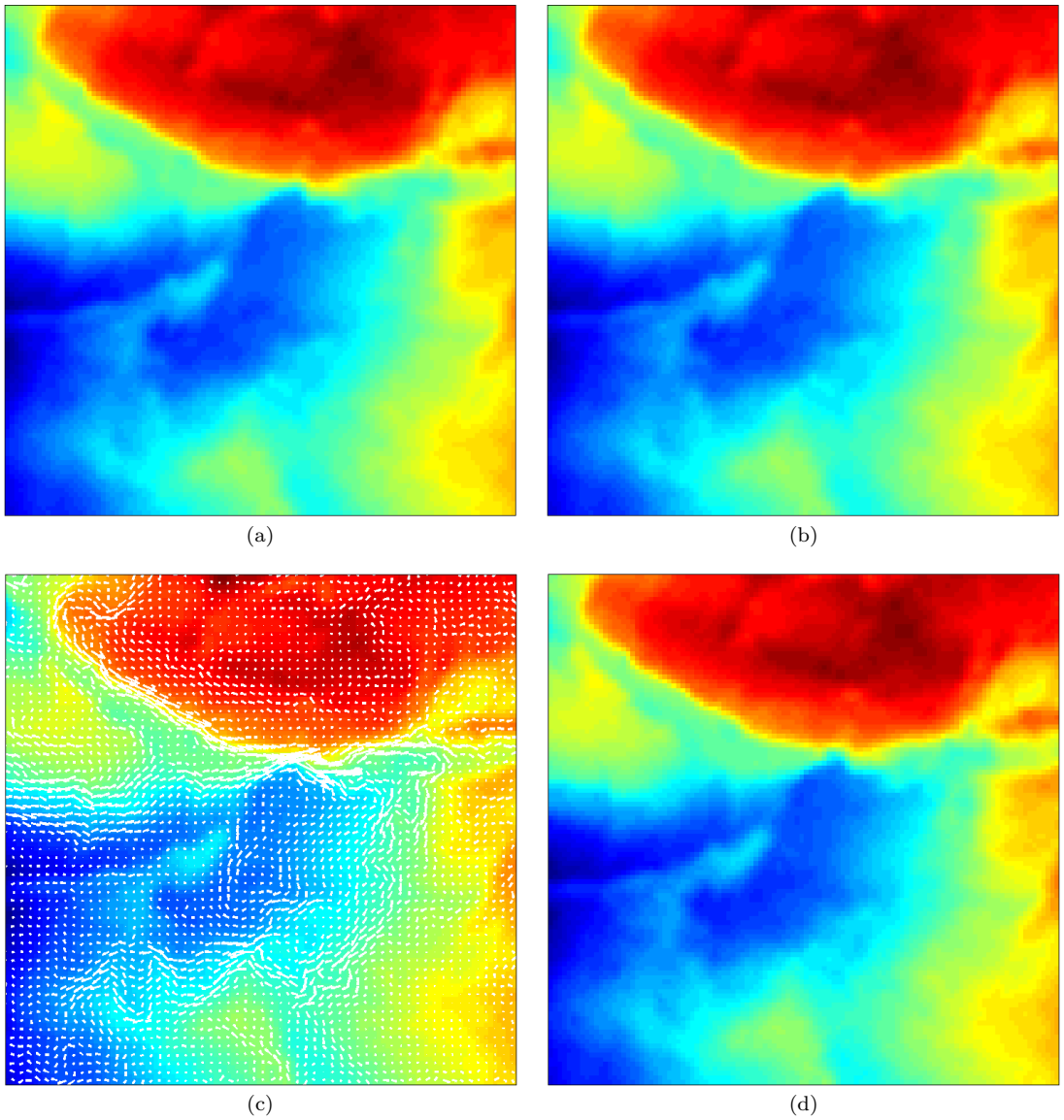


Figure 3.9: Sea Surface Temperature Flow – Images (a) and (b) represent two different time instances (one hour apart from each other) of an image sequence of sea surface temperature along the coast of Oregon (USA) on August, 1, 2002. Six different optical flow algorithms which include conservation of intensity data fidelity are employed on images (a) and (b). The images (c) and (d) show the reconstructed flow fields using the stream function method with the regularization terms R_2 and R_3 respectively. Also images (e) and (f) show the computed flow field from the potential function method with the regularizations R_2 and R_3, and images (g) and (h) show the computed optical flow field using the u-v approach with R_2 and R_3. As can be seen here, only the stream function formulation captures the gyres very well, but none of the potential or u-v methods captures any of these gyres presented on data.



(a) (b)

(c) (d)

Figure 3.10: GOCI Flow – Images (a) and (b) show two time instances, with the temporal resolution of 10 minutes, of products movements in the seas of South Korea. Image (c) displays the computed flow field from the stream function method with R_2 regularization and the image (d) shows the reconstructed second image by evolving the first image in (a) forward in time under the evolution model (3.2) with the computed flow field as shown in image (c). The mean relative error is 2.21 % for the stream function formulation and 5.32 % for the $u - v$ formulation.

3.4 Another Advantage of Stream Function Method over the u-v Method

According to the results from the synthetic data sets as well as the real oceanic data sets, the stream function method outperforms the u-v method in fluid systems. The stream function method provides another advantage when we regularize the optical flow problems by imposing the prior knowledge of the solution (scientific priors), which we explain in detail in Chapter 4. When we impose the prior knowledge, it is important to identify the correct formulation of the problem. We may have problems in terms of the u-v formulation or the stream function formulation. In the u-v method the prior knowledge is imposed on the components of the flow, but in the stream function formulation the prior knowledge can be directly imposed on the flow. For instance, as in [37] if the expected flow is sparse, we use L^1 norm regularization to reconstruct the flow. In the u-v method, the commonly used regularization term [60] to reconstruct sparse flow is

$$R(u,v) = \int_\Omega |u| + |v| \, d\Omega. \tag{3.26}$$

The Euler-Lagrange equations for the regularization term in Eq.(3.26) are obtained as

$$\begin{aligned} \frac{u}{\sqrt{u^2+\epsilon}} &= 0 \text{ and} \\ \frac{v}{\sqrt{v^2+\epsilon}} &= 0, \end{aligned}$$

where ϵ is a small real number to be chosen. The first equation involves only u and the second equation involves only v, so they are not coupled. Since the two equations are independent of each other, the optimal solution under this regularization tends to reconstruct the component wise sparse flow fields irrespective of the data term. That

is, the resulting flow field may have zeros in some places of the u component and zeros in some places of the v component. Even though both the u and v components have zeros, there is no guarantee that the zeros in both components are at the same place and hence the flow field may not be sparse. On the other hand, if we use the stream function formulation, the sparsity of the flow would be measured by the total variations of the stream function ψ. The appropriate total variation regularization term is

$$R(\psi) = \int_{\Omega} |\nabla_H \psi| \, d\Omega. \tag{3.27}$$

The Euler-Lagrange equation for the Total variation regularization term is obtained as

$$\nabla \cdot \left(\frac{\nabla_H \psi}{|\nabla_H \psi|} \right) = 0 \tag{3.28}$$

and it consists of both components $-\psi_y$ and ψ_x. This formulation allows us to impose regularity on the flow to reconstruct sparse flow. The Euler-Lagrange equation corresponding to the sparsity regularization term in Eq.(3.28) is the same as the potential function formulation since $|\nabla_H \psi|$ and $|\nabla \psi|$ are the same.

The above explanation demonstrates an additional advantage of the stream function formulation over the u-v method when imposing prior knowledge in optical flow problems. In the Chapter 4, a detailed explanation of imposing prior knowledge via the regularization term will be included.

Chapter 4

Regularization of Optical Flow Problem

The computation of optical flow involves minimizing functionals which leads to solving an inverse problem. Generally, the optical flow problem is an ill-posed problem because the data comes with noise and discretization errors. Recall that, according to Hadamard [49], a problem $Au = z$ is called well-posed, if there is a solution to the problem and the solution is unique and continuous with respect to the perturbations of z. If at least one of these conditions is not satisfied, then the problem is ill-posed. In optical flow problems, especially the perturbations in input data may cause the instability of the solution. One of the common approaches to over come these kind of difficulties is to regularize the ill-posed problem. The main reason to regularize the ill-posed problem is to enforce the well-posedness. That is, we want to make sure that there exists a unique solution which is stable with respect to the perturbations in the input data. The one important thing on which we concentrate when solving the regularized problem is whether the regularized solution satisfies the original problem. We can achieve this by seeking a solution to the regularized problem that approximately minimizes the data fidelity for small quantities of the regularization parameter. When

seeking the solution of the optical flow problem, both the left hand side and the right hand side depend on the input data and this may cause the well-posedness of the problem even after the regularization. This incident can often be seen in the optical flow problems.

The other hidden advantage of regularization is to incorporate prior knowledge of the solution to the Mathematical problem via the regularization term. This is a great advantage in optical flow problems as it is hard to achieve the well-posedness after the regularization. For instance, both data fidelities in Eq.(2.7) and Eq.(3.4) do not contribute any prior knowledge on the estimated flow but do impose the prior knowledge on the data. If we think of adding the regularization term introduced in Horn and Schunck to the above data fidelities, the minimization problem will take into account that the expected flow would be smooth.

4.1 A Simple Explanation about Regularization

As we mentioned earlier, the first goal of regularizing an optimization problem is to ensure the well-posedness so that the solution of the problem satisfies the properties of existence, uniqueness and stability. When we have to solve an ill-posed inverse problem, we reformulate the problem as an optimization problem and then regularize it. In this section we will illustrate how to achieve well-posedness using linear inverse problems.

Example 4.1.1. Consider the problem [12]

$$\begin{pmatrix} 2 \\ 1 \end{pmatrix} x = \begin{pmatrix} 4 \\ 2.1 \end{pmatrix} \tag{4.1}$$

and sove for x.

This is an overdetermined linear system which has no solution because there does not exist a solution x such that $x = 2$ and $x = 2.1$. Since the solution does not satisfy the property of existence, the problem is ill-posed. However, we can reformulate the problem as an optimization problem in the form of least squares as

$$\arg\min_{x} \left\| \begin{pmatrix} 2 \\ 1 \end{pmatrix} x - \begin{pmatrix} 4 \\ 2.1 \end{pmatrix} \right\|_2^2 . \tag{4.2}$$

The solution for the new problem is $x = 2.02$ and it is unique. The reformulation of the problem leads to a unique solution even though the original problem does not have a solution. However, the uniqueness property of the solution is not often achievable. The following example in [61] will explain the difficulty of achieving a unique solution.

Example 4.1.1. Consider a system of two equations [61]

$$\begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 3 \\ 3 \end{pmatrix} . \tag{4.3}$$

to be solved for x_1 and x_2.

This problem does not have any solution as one equation of the system implies that $0 = 3$. Therefore, the problem does not satisfy the existence property and hence is ill-posed. Now we reformulate the problem as a minimization problem just like we did in the previous example. Then the new problem becomes

$$\arg\min_{x_1,x_2} \left\| \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} 3 \\ 3 \end{pmatrix} \right\|_2^2 . \tag{4.4}$$

The problem in Eq.(4.4) is minimized when $x_1 + 2x_2 = 3$. Since there are infinitely many solutions which satisfy this constraint, the reformulated problem has infinitely many solutions as well. Now in this example, the original problem does not have any

solutions whereas the reformulated problem has infinitely many solutions. Note that these solutions are obtained when the distance between $\begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ and $\begin{pmatrix} 3 \\ 3 \end{pmatrix}$ is 3, which should be zero. The next step is to select an appropriate solution among the infinitely many choices. To obtain the unique solution, we must modify the problem again which we accomplish by including an additional constraint to the problem. In other words we must define a new characteristic of the solution in order to identify the desired solution. This new characteristic is defined using the prior knowledge of the expected solution; this notion is often called the *scientific prior*.

Now we are going to add another term to the problem in Eq.(4.4). For instance, if we wish to have a solution on $x_1 + 2x_2 = 3$ which is closest to the origin, then the regularized problem of Eq.(4.4) becomes

$$\arg\min_{x_1,x_2} \left\| \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} 3 \\ 3 \end{pmatrix} \right\|_2^2 + \left\| \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\|_2^2 . \tag{4.5}$$

The solution that gives the smallest euclidean distance from the origin to the line $x_1 + 2x_2 = 3$ is $\begin{pmatrix} 0.6 \\ 1.2 \end{pmatrix}$. Therefore, the introduction of the regularization term in l^2 norm ensures the well-posedness of the solution and hence the new problem has a unique solution. Fig. 4.1 is a graphical representation of approaching the solution in the sense of l^2 norm. The closest distance from the origin to the line is shown by the intersection point of the line and the ϵ ball l^2 norm. The closest distance between line and the origin is given by the radius of the circle.

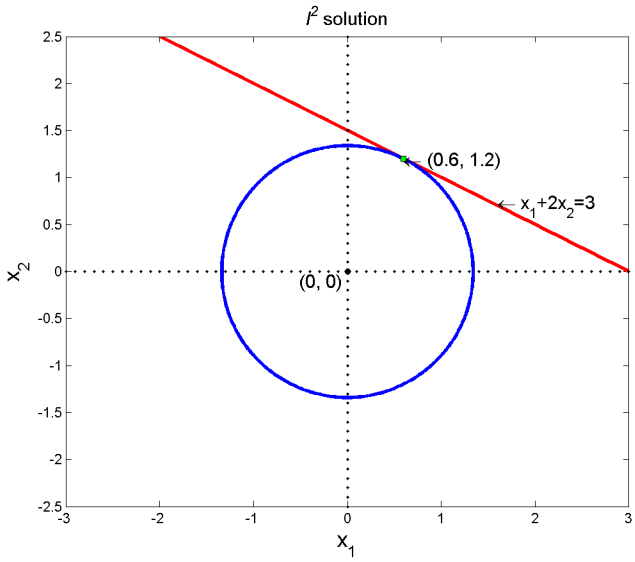


Figure 4.1: l^2 solution – Image shows the graphical representation of the solution after introducing the l^2 regularization.

Another way to regularize the problem is to find the closest point on the line to the origin under the l^1 norm.

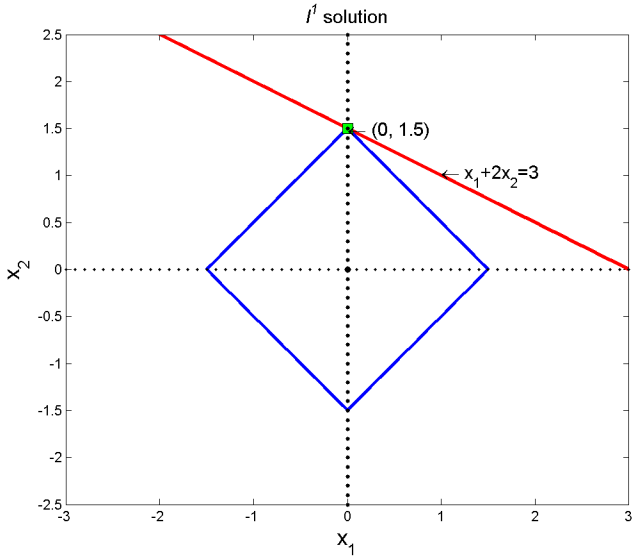


Figure 4.2: l^1 solution – Image shows the graphical representation of the solution under the l^1 regularization.

With the l^1 norm distance, the problem can be rewritten as

$$\arg\min_{x_1,x_2} \left\| \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} 3 \\ 3 \end{pmatrix} \right\|_2^2 + \left\| \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\|_1 . \tag{4.6}$$

The l^1 norm solution will be on the intersection of the line $x_1 + 2x_2 = 3$ and one of the axes. Thus the solution is $\begin{pmatrix} 0 \\ 1.5 \end{pmatrix}$ because it minimizes $\mid x_1 \mid + \mid x_2 \mid$. Fig. 4.2 shows the graphical representation of the solution which we obtained from the l^1 regularization. Moreover, we can use ∞-norm and then the problem becomes

$$\arg\min_{x_1,x_2} \left\| \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - \begin{pmatrix} 3 \\ 3 \end{pmatrix} \right\|_2^2 + \left\| \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \right\|_\infty . \tag{4.7}$$

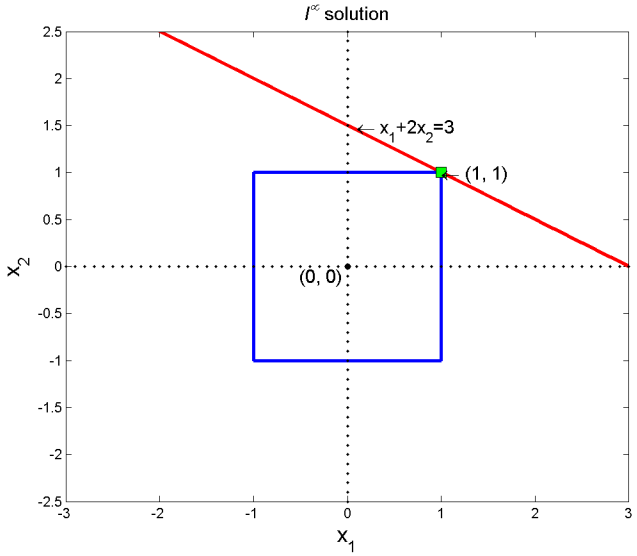


Figure 4.3: l^1 solution – Image shows the graphical representation of the solution under the l^1 regularization.

The solution is $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ which minimizes the distance to the line from the origin under the l^∞ norm. Fig. 4.3 shows the l^∞ regularize solution for the problem. The

closest distance to the origin is the intersection point of ϵ-ball of the l^∞ norm and the line $(1,1)^T$ is shown in a green marker point. In all three cases, i.e. under l^1, l^2 and l^∞ regularization, we have a unique solution, but they are not the same. The reason for the different solutions is each of these solutions depends on the new characteristic that we introduced. That is, the scientific prior causes the desired solution.

4.2 Scientific Priors

Toward our discussion of designing scientific prior information into our functional, we first review some well known theories of inverse problems [11, 12, 49]. Consider a simple linear system which we often solve in linear algebra

$$Au = z, \tag{4.8}$$

where $A \in \mathbb{R}^{m\times n}$, $u \in \mathbb{R}^n$ and $z \in \mathbb{R}^m$. When A and u are given, it is called a forward problem. When A and z are available and the system is solved for u, it is called an inverse problem. For an inverse problem, a unique solution can be obtained by

$$u = A^{-1}z, \text{ if } A^{-1} \text{ exists.}$$

If A is not invertible, then the system may have infinitely many solutions or no solution. In either case, as we explained in Sec. 4.1, it is standard to reformulate the problem as a minimization problem in such a way that from infinitely many solutions we can select or emphasize a desired solution. After reformulating the problem in the least square sense, it can be written as

$$u_{LS} = \arg\min_u \|Au - z\|_2^2. \tag{4.9}$$

In the modified problem, we find u_{LS} in such a way that we minimize the distance between Au and z in the Euclidean perspective. As we have seen in Sec. 4.1, the above reformulation makes a unique solution or infinitely many solutions even when there are no solutions to the original problem. Therefore, the above formulation in Eq.(4.9) is the first step to achieving a unique solution when the original problem is ill-posed. This alternative approach of solving the original inverse problem ensures at least one solution to the original problem. The second step is to reach a unique solution when there exits more than one solution. We achieve the uniqueness by including an additional constraint to the reformulated problem in Eq.(4.9) to select the appropriate solution.

In addition to the existence and uniqueness of a solution, the regularization enforces the stability of the solution. If the matrix A in Eq.(4.8) is ill-conditioned, then the small perturbations of z may give large perturbations in the solution u. When we applied it to the real world cases, the data may contain errors from measurements and then from the discretization. Let b be the true data and e be the errors accompanying the measured data, where $b, e \in \mathbb{R}^m$. Then taking the measurement errors into account the right hand side z, which is the measured data of the Eq.(4.8), can be considered as $z = b + e$. Now the original problem becomes

$$Au = b + e. \tag{4.10}$$

4.2.1 An Illustrative Example

The following example in [12] illustrates the stability of an ill-posed problem subject to the small perturbations of the measured data z. Consider the linear system

$$A = \begin{pmatrix} 0.16 & 0.10 \\ 0.17 & 0.11 \\ 2.02 & 1.29 \end{pmatrix} \text{ and } u = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \text{ then } b = \begin{pmatrix} 0.26 \\ 0.28 \\ 3.31 \end{pmatrix}.$$

When we introduce a small perturbation $e = (0.01, -0.03, 0.02)^T$ to the true solution b, the right hand side becomes $z = (0.27, 0.25, 3.33)^T$. The corresponding least square solution is

$$u_1 = \begin{pmatrix} 7.04 \\ -8.40 \end{pmatrix} \text{ and the residual norm, } \|Au_1 - z\|_2 = 0.022.$$

Considering the small residuals close to 0.022, we can obtain two other alternative solution:

$$u_2 = \begin{pmatrix} 1.65 \\ 0 \end{pmatrix} \text{ and the residual norm, } \|Au_1 - z\|_2 = 0.031 \text{ and}$$

$$u_3 = \begin{pmatrix} 0 \\ 2.58 \end{pmatrix} \text{ with the residual norm, } \|Au_1 - z\|_2 = 0.036.$$

All three solutions lead to small residuals, but the estimated solutions have deviated from the true solution $(1, 1)^T$. This phenomenon occurs because the matrix A is ill-conditioned. Thus, the example illustrates that even a small error in the input data of an ill-posed problem causes the solution we seek to be near the true solution. Therefore, we can see that in general, an invertible A and an error free right hand side will provide a unique solution. However, if A is not invertible and right hand side

is contaminated with errors, we have to put in greater effort to solve the problem. In reality, most of the Science and Engineering applications are the latter case.

One possible way of finding a solution of ill-posed systems is to apply the Singular Value Decomposition (SVD) on the operator A and then solve the system with further modifications. As the first step of this approach, we explain the derivation of the SVD of the matrix A using the following theorem.

Theorem 4.2.1. [62] *If A is a real m-by-n matrix, then there exist orthogonal matrices $U = (u_1, u_2, \cdots, u_m) \in \mathbb{R}^{m\times m}$ and $V = (v_1, v_2, \cdots, v_n) \in \mathbb{R}^{n\times n}$ such that*

$$A = U\Sigma V^T,$$

where $\Sigma = diag\,(\sigma_1, \sigma_2, \cdots, \sigma_p) \in \mathbb{R}^{m\times n}$ with $p = \min\{m, n\}$. Here $\sigma_i \geq 0$ for $i = 1, 2, ..., p$, are called the singular values of A.

In this theorem, it is assumed that the singular values $\sigma_1, \sigma_2, \cdots, \sigma_p$ are in descending order. That is $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0$. Now the matrix A can be represented by an alternative form using the singular value decomposition as

$$\begin{aligned} A &= \begin{bmatrix} u_1 & u_2 \ldots & u_m \end{bmatrix} \begin{bmatrix} \sigma_1 & & & \\ & \sigma_2 & & \\ & & \ddots & \\ & & & \sigma_p \end{bmatrix} \begin{bmatrix} v_1^T \\ v_2^T \\ \vdots \\ v_n^T \end{bmatrix} \\ &= \sum_{k=1}^{p} \sigma_k u_k v_k^T. \end{aligned}$$

Since the matrices U and V are orthogonal, the inverse of A can be obtained as

$A^{-1} = V\Sigma^{-1}U^T$. In terms of singular values, we can obtain A^{-1} as

$$A^{-1} = \sum_{k=1}^{p} \frac{1}{\sigma_k} v_k u_k^T.$$

Then the solution for the system in Eq.(4.10) can be obtained as

$$\begin{aligned} u &= \sum_{k=1}^{p} \frac{v_k u_k^T}{\sigma_k}(b+e) \\ &= \sum_{k=1}^{p} \frac{u_k^T b}{\sigma_k} v_k + \sum_{k=1}^{p} \frac{u_k^T e}{\sigma_k} v_k. \end{aligned}$$

According to this solution, when the the singular values are zero or close to zero, the errors in the right hand side will be amplified. This amplification diverts the solution from the actual solution; hence the direct SVD method is not appropriate. To overcome this kind of amplification of the right hand side of the system, we can introduce a threshold for the singular values so that we can eliminate the singular values corresponding to the amplifications. This Method is called the Truncated Singular Value Decomposition (TSVD). If the threshold is at r^{rh} singular value, the solution for the system in Eq.(4.10) is obtained as

$$u = \sum_{k=1}^{r} \frac{u_k^T (b+e)}{\sigma_k} v_k.$$

The solution from the TSVD method with $z = b + e$ can be obtained by introducing a filter factor to the SVD solution as

$$u = \sum_{k=1}^{p} \varphi_k \frac{u_k^T z}{\sigma_k} v_k, \tag{4.11}$$

where the k^{th} filter factor φ_k is given as

$$\varphi_k = \begin{cases} 1, & \text{if} \quad 1 \leq k \leq r \\ 0, & \text{if} \quad r < k \leq p. \end{cases}$$

However, in real applications we have to deal with the large-scale matrices. Since the computation of Singular Value Decomposition of A is numerically costly, this is less than ideal. For instance, if we have to compute the optical flow field from images of size 300×400, then the size of the resulting matrix A would be $120,000 \times 120,000$. Since TSVD method is inefficient for large-scale problems, we must have an alternative approach to solve this problem. The most commonly used method to solve ill-posed problems in the literature is the Tikhonov regularization method [63] which was introduced by Andrey Tikhonov in 1963.

Definition 6. *[12] Consider the ill-posed problem $Au = z$ in Eq.(4.8). Then the Tikhonov solution u_α to the problem is*

$$u_\alpha = \arg\min_u \|Au - z\|_2^2 + \alpha \|u\|_2^2, \tag{4.12}$$

where α is the regularization parameter.

This is done by adding a new term to the reformulate problem in Eq.(4.9) such as a new constraint which is equivalently defining a new characteristic of the solution. The new formulation helps to achieve the existence, uniqueness and stability of the solution. The Tikhonov regularization method can be generalized by adding different terms instead of $\|u\|_2^2$. In real world applications, if we introduce the new term using the prior knowledge of the solution, we call this notion the scientific prior of this simple linear problem.

When we determine the solution from the formulation in Eq.(4.12), first we have to obtain the corresponding Euler-Lagrange equation and then solve it for u. After

solving the Euler-Lagrange equation, the solution for u may be obtained as

$$u = \left(A^T A + \alpha I\right)^{-1} A^T z. \tag{4.13}$$

Taking the SVD of A and choosing a suitable regularization parameter α, we can re-write the solution for u as

$$u = \sum_{k=1}^{p} \frac{\sigma_k}{\sigma_k^2 + \alpha} (u_k^T z) v_k.$$

Rearranging the terms, the solution can be expressed in terms of a filter factor as

$$u = \sum_{k=1}^{p} \varphi_k \frac{u_k^T z}{\sigma_k} v_k, \tag{4.14}$$

where the k^{th} filter factor φ_k is given as

$$\varphi_k = \frac{\sigma_k^2}{\sigma_k^2 + \alpha}, \text{ for } k = 1, 2, ..., p.$$

According to Eq.(4.11) and Eq.(4.14), the solution from both TSVD and the Tikhonov regularization schemes are both a specific case of a general regularization framework with different filter factors. In the TSVD method, we eliminate singular values which are zero or close to zero to avoid the amplification of errors in the solution. Whereas in the Tikhonov regularization method, we approach the same goal by adding a small positive real number to the singular values which are zero or close to zero. This tells us that even though the basic approaches are completely different, there is still a connection between these two schemes. However, the Tikhonov regularization method out performs the TSVD method in large-scale problems. In addition to this, in the Tikhonov regularization method we can introduce different filter factors for different characteristics. The ability to make such modifications makes

the method stronger.

There are two main reasons to introduce a regularization term to an ill-posed problem. The first reason is to enforce the well-posedness to the problem and the second reason is to emphasize the prior knowledge of the desired solution. The solution from the Tikhonov regularization would gives us the minimizer of the residual norm under the weighted minimum of the solution norm. The selection of the best regularization parameter plays an important role in the regularized solution and we will include a detailed discussion in Sec. 4.6. In general, the Tikhonov regularization can be extended for any regularization term by writing Eq.(4.12) as

$$\arg\min_{u} \|Au - z\|_2^2 + \alpha \|Lu\|_2^2, \tag{4.15}$$

where L is a preferred regularization operator. For instance, L may be the gradient operator.

In addition to enforcing the prior knowledge of the flow via the regularization term, the prior knowledge of the data can be applied to the problem at the beginning of the construction of the model by choosing different operators for A in the data fidelity. Analogously, building scientific prior information into functionals allows our inverse problem solutions for vector fields to emphasize expected physics.

Now we will discuss various operators and regularization terms to emphasize expected scientific prior information. In this discussion without loss of generality, we list the data terms and the regularization terms in terms of stream function rather than u and v for our convenience.

In the nominal optical flow algorithm, Horn and Schunck assumed conservation of image brightness locally for rigid body motion. For the entire domain Ω with time step t, the conservation of image brightness I is emphasized by

$$E(\psi) = \int_{\Omega} \left(I_t - I_x\psi_y + I_y\psi_x\right)^2 \, d\Omega, \tag{4.16}$$

where ψ is the corresponding stream function for the motion. However, later researchers interested in fluid motion assumed that the image brightness I behaves according to the continuity equation over time in order to allow for divergent flow fields. Therefore authors in [29] proposed the data fidelity term

$$E(\psi) = \int_{\Omega} \left(I_t - I_x\psi_y + I_y\psi_x - I\psi_{yx} + I\psi_{xy}\right)^2 \, d\Omega. \tag{4.17}$$

Also the authors in [41] improved the data fidelity by imposing the constancy of spatial brightness gradient, instead of brightness constancy, with the following,

$$E(\psi) = \int_{\Omega} \left(I_{xt} - I_{xx}\psi_y + I_{xy}\psi_x\right)^2 + \left(I_{yt} - I_{yx}\psi_y + I_{yy}\psi_x\right)^2 \, d\Omega. \tag{4.18}$$

Further, researchers in [64] have combined the data fidelities Eq.(4.16) and Eq.(4.18) together with a non-negative parameter λ. The resulting data fidelity is obtained as

$$\begin{aligned} E(\psi) = \int_{\Omega} \left(I_t - I_x\psi_y + I_y\psi_x\right)^2 \, d\Omega + \lambda \int_{\Omega} \left(I_{xt} - I_{xx}\psi_y + I_{xy}\psi_x\right)^2 \, d\Omega \\ + \lambda \int_{\Omega} \left(I_{yt} - I_{yx}\psi_y + I_{yy}\psi_x\right)^2 \, d\Omega. \end{aligned} \tag{4.19}$$

In this data fidelity, either image brightness is constant or the gradient of the image brightness is constant, by adjusting λ to emphasize either the expected outcome or the underlying physics. Through this kind of approach, the notion of modeling the physics that is creating the scenes evolution in the movie-data is designed into the functional.

In addition to the data fidelities, it is important to regularize the energy functional as understood by Andrey Tikhonov as shown in [65]. We do so to extract important information from ill-posed problems, both for functional analysis and for optimization theoretic reasons of well-posedness, and to further emphasize expected physics. For instance, if the resulting flow is expected to be sparse, then the regu-

larization term that emphasizes sparsity is the total variation of the stream function. The regularization term can be written as

$$R(\psi) = \int_{\Omega} \mid \bigtriangledown \psi \mid d\Omega. \tag{4.20}$$

If, however, the flow field is expected to be smooth, then the appropriate regularization is the Horn and Schunck type regularization,

$$R(\psi) = \int_{\Omega} (\psi_{xx}^2 + \psi_{yy}^2 + \psi_{xy}^2 + \psi_{yx}^2) d\Omega. \tag{4.21}$$

On the other hand, if we impose the regularity on flow components, we would use

$$R(\psi) = \int_{\Omega} (\psi_{x}^2 + \psi_{y}^2) d\Omega. \tag{4.22}$$

According to the above explanation, we can develop different data terms and regularization terms using the known physics of the data and the expected flow. We can construct various algorithms according to the prior knowledge of systems being imaged. Hence, applying a suitable algorithm, the motion field of the observed system can be determined.

4.3 Comparison of Different Data Terms and Regularization Terms

Even after the regularization of optical flow problems, they are rarely well-posed as the the data fidelity depends on the input data. For this reason, regularizing the optical flow mainly focuses on imposing the scientific priors. Our next goal is to compare two data fidelities in Eq.(4.16) and Eq.(4.17) with six different regularization terms on our synthetic data sets. Various ways of selecting data terms and regularizing optical

flows for rigid body motion are explained in [41,66] and for the fluid motions in [23,24]. Except for the Horn-Schunck regularization (R_2), the other five regularization terms do not appear regularly in the optical flow literature. The reason for this is that in the classical optical flow approach the flow components u and v are directly regularized. But in our approach, we impose the the regularity directly on the stream function. In the next step, we explain about the six regularization terms with the corresponding Euler- Lagrange equations as we use them to minimize the energy functional.

The first regularization term is defined as

$$R_1(\psi) = \int_\Omega \psi^2 + \psi_x^2 + \psi_y^2 + \psi_{xx}^2 + \psi_{yy}^2 \, d\Omega. \tag{4.23}$$

We name this as the **convexity** regularization term and the gradient of $R_1(\psi)$ can be obtained using the theory in Sec. 2.0.1 as

$$GR_1(\psi) = (B_1 + B_1^*)\psi, \quad \text{where} \quad B_1 = I + D_x^* D_x + D_y^* D_y + D_{xx}^* D_{xx} + D_{yy}^* D_{yy}.$$

The regularization term R_1 introduces the coercivity and strictly convexity which we defined in Sec. 2.1, to the problem. These properties lead the problem to a unique solution in $H^2(\Omega)$. Even though the uniqueness is expected, it is rarely achieved in optical flow problems and imposing the scientific prior is the main advantage of this regularization term. Next, we consider the Horn-Schunck regularization term which emphasizes the smoothness of the flow:

$$R_2(\psi) = \int_\Omega \psi_{xx}^2 + \psi_{xy}^2 + \psi_{yx}^2 + \psi_{yy}^2 \, d\Omega. \tag{4.24}$$

We name R_2 as the **smoothness** regularization term. As we already know, the gradient of the $R_2(\psi)$ is

$$GR_2(\psi) = (B_2 + B_2^*)\psi, \quad \text{where} \quad B_2 = D_{xx}^* D_{xx} + D_{yy}^* D_{yy} + D_{xy}^* D_{xy} + D_{yx}^* D_{yx}.$$

In the third regularization term, we minimize the velocity components u and v in the sense of u-v formulation under $L^2(\Omega)$. We can express the term

$$R_3(\psi) = \int_\Omega \psi_x^2 + \psi_y^2 \, d\Omega, \tag{4.25}$$

and name it as **velocity component** regularization term and the gradient of $R_3(\psi)$ as

$$GR_3(\psi) = (B_3 + B_3^*)\psi, \quad \text{where} \quad B_3 = D_x^* D_x + D_y^* D_y.$$

The next regularization term was introduced in [34] for the optical flow computation in the **engineering strain tensor** formulation. It can be written as

$$R_4(\psi) = \int_\Omega \left(\psi_{xx} - \psi_{yy}\right)^2 + \left(\psi_{xy} + \psi_{yx}\right)^2 \, d\Omega + \int_\Omega \psi_{yxx}^2 + \psi_{xyy}^2 \, d\Omega. \tag{4.26}$$

The gradient of $R_4(\psi)$ may be obtained as

$$GR_4(\psi) \;=\; (B_4 + B_4^*)\psi, \quad \text{where}$$

$$B_4 = (D_{xx} - D_{yy})^*(D_{xx} - D_{yy}) + (D_{xy} + D_{yx})^*(D_{xy} + D_{yx}) + D_{yxx}^* D_{yxx} + D_{xyy}^* D_{xyy}.$$

The regularization term R_5 also comes from the literature in [34], which represents the **second-order div-curl** regularization. This term is suitable for a non-hyperbolic flow field. However, this term imposes the smoothness on the flow and can be written as

$$R_5(\psi) = \int_\Omega (\psi_{xx} + \psi_{yy})^2 + (\psi_{xy} - \psi_{yx})^2 \, d\Omega \tag{4.27}$$

The gradient of $R_5(\psi)$ is derived as

$GR_5(\psi) = (B_5 + B_5^*)\psi$, where

$$B_5 = (D_{xx} + D_{yy})^*(D_{xx} + D_{yy}) + (D_{xy} - D_{yx})^*(D_{xy} - D_{yx})$$

The last regularization term we introduce for the purpose of comparison is given as

$$R_6(\psi) = \int_\Omega (\psi_{xx} - \psi_{yy})^2 + (\psi_{yx} - \psi_{xy})^2 \, d\Omega. \tag{4.28}$$

This regularization term provides the smoothness to the flow and appropriate for non-rotational flow. The gradient of $R_6(\psi)$

$GR_6(\psi) = (B_6 + B_6^*)\psi$, where

$$B_6 = (D_{xx} - D_{yy})^*(D_{xx} - D_{yy}) + (D_{yx} - D_{xy})^*(D_{yx} - D_{xy}).$$

The above six regularization terms in Eqs.(4.23) - (4.28) can be expressed in terms of u and v. The revised regularization terms are shown in first column of the Table 4.1. In u-v formulation, there are two Euler-Lagrange equations for each regularization term and they are shown in the second column of the Table 4.1.

Table 4.1: Stream function formulation regularization terms are represented in u-v formulation. Six different regularization terms in Eqs.(4.23) - (4.28) are written in u-v formulation and listed in the first column. The second column shows two Euler-Lagrange equations for for each regularization term.

Regularization term	First variations w.r.t. u and v
$R_2(u,v) = \int_\Omega \left(u_x^2 + u_y^2 + v_x^2 + v_y^2\right)\, d\Omega$	$\frac{\delta R_2}{\delta u} = 2\left(u_{xx} + u_{yy}\right)$ $\frac{\delta R_2}{\delta v} = 2\left(v_{xx} + v_{yy}\right)$
$R_3(u,v) = \int_\Omega \left(u^2 + v^2\right)\, d\Omega$	$\frac{\delta R_3}{\delta u} = 2u$ $\frac{\delta R_3}{\delta v} = 2v$
$R_4(u,v) = \int_\Omega \left(u_y + v_x\right)^2 + \left(u_x + v_y\right)^2\, d\Omega$ $+ \int_\Omega \left(u_{xx}^2 + v_{yy}^2\right)\, d\Omega$	$\frac{\delta R_4}{\delta u} = 2\left(u_{xx} + u_{yy} + v_{xy} - v_{yx}\right)$ $\frac{\delta R_4}{\delta v} = 2\left(-u_{xy} + u_{yx} + v_{xx} + v_{yy}\right)$
$R_5(u,v) = \int_\Omega \left(u_x + v_y\right)^2 + \left(u_y - v_x\right)^2\, d\Omega$	$\frac{\delta R_5}{\delta u} = 2\left(u_{xx} + u_{yy} - v_{xy} + v_{yx}\right)$ $\frac{\delta R_5}{\delta v} = 2\left(u_{xy} - u_{yx} + v_{xx} + v_{yy}\right)$
$R_6(u,v) = \int_\Omega \left(u_x + v_y\right)^2 + \left(u_y + v_x\right)^2\, d\Omega$	$\frac{\delta R_6}{\delta u} = 2\left(u_{xx} + u_{yy} + v_{xy} + v_{yx}\right)$ $\frac{\delta R_6}{\delta v} = 2\left(u_{xy} + u_{yx} + v_{xx} + v_{yy}\right)$

4.4 Mean Angular Error (MAE)

In Chapter 3, we compare the results from the stream function approach and the u-v approach with conservation of image brightness and the continuity equation data fidelities and with R_2 and R_3 regularization terms by considering the qualitative differences of reconstructed flow fields. The comparison of the quality of thousands of vectors, however, is not a rigorous approach without a proper measurement. Hence we need a quantitative measurement to compare our methods. In [67], the authors introduced an error measurement to compare reconstructed velocity fields using the average angular error between the computed flow and the actual flow. If we denote the computed flow and the true flow as $z_C = \langle u_C, v_C \rangle$ and $z_T = \langle u_T, v_T \rangle$ respectively,

the angular error matrix for each pixel on the image is obtained as

$$\theta = \cos^{-1}\left(\frac{z_c \cdot z_T}{|z_c||z_T|}\right). \tag{4.29}$$

Then the average angular error is obtained by taking the average of θ over the number of pixels. According to this formulation, flow vectors with magnitudes close to zero produce large errors while flow vectors with higher magnitudes produce small errors. To overcome these drawbacks, the authors in [68] modified the computation of average angular error and it was named Mean Angular Error (MAE). For MAE, a third component δ is added to both the computed and the true velocity fields. Then the computed and the true flow vectors become $z_C = \langle u_C, v_C, \delta\rangle$ and $z_T = \langle u_T, v_T, \delta\rangle$ respectively. In this case, $\delta = 1$. It affects flow with small magnitudes to reduce the error, but does not affect flows with large magnitudes. Therefore, δ provides a threshold to the errors near small flows. Now we can use the formula in Eq.(4.29) to compute the angular error θ. Then the Mean angular error is obtained by taking the average over the number of pixels.

For instance, consider the computed flow field from the saddle images with the regularization parameter $\alpha = 10^{-12}$. In this example, we use the stream function formulation with conservation of energy data fidelity and the smoothness regularization term. First we compute the angular error for each pixel and then average the error. Fig. 4.4 shows a histogram of the angular errors. According to the histogram, when the angular error is small, the frequency is high, whereas the angular error is large, the frequency is low. The resulting mean angular error is 0.1843°

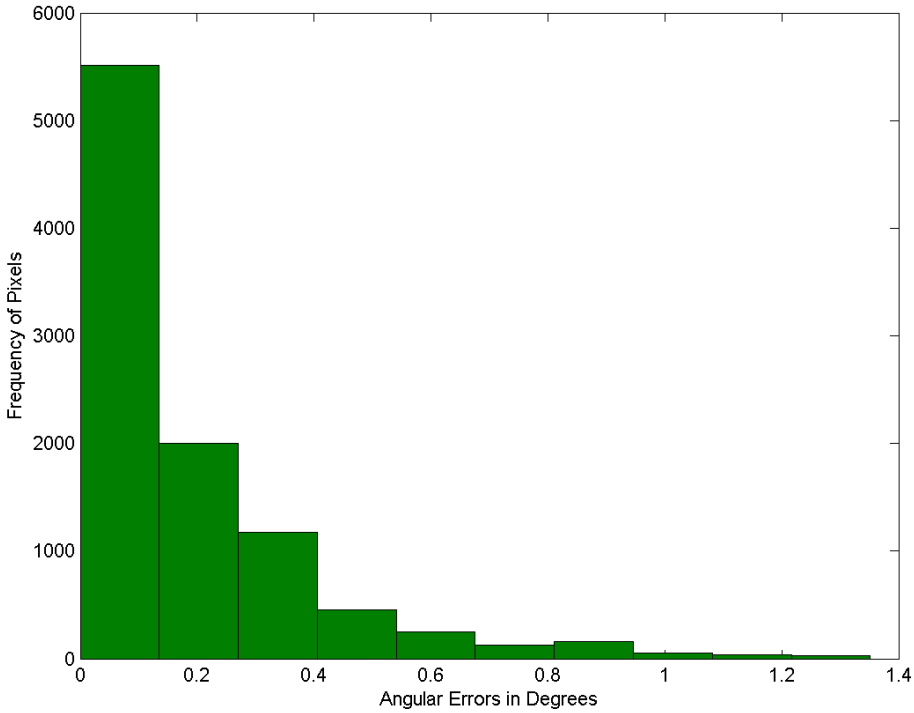


Figure 4.4: Histogram of Angular Errors – The histogram shows the frequency of the points with respect to the angular error for the computed flow field for the saddle images. The flow field was computed using the stream function formulation with conservation of intensity data fidelity and the smoothness regularization term.

4.5 Results and Error Analysis

In this section, we compare the six regularization terms which we introduced in Eqs.(4.23)-(4.28) and the two data fidelities Eq.(4.16) and Eq.(4.17). Combinations of two different data terms and six different regularization schemes lead to twelve different optical flow algorithms. Further we combine the regularization term R_1 with R_2 and R_3 for both data terms to enforce the uniqueness of the solution in stream function formulation. In addition to the stream function formulation, we use two data terms and five regularization terms except R_1 in the u-v formulation and to ensure the uniqueness in the u-v formulation, we combine R_2 and R_3 as there does not exist a u-v formulation for R_1. All these resulting algorithms are used to reconstruct flow

fields on the three synthetic data sets which we introduced in Sec. 2.3. For the quantitative comparison of results, we compute the modified MAE in Sec. 4.4 on each of the reconstructed flows.

Table 4.2: Combining two data terms which represents the Continuity Equation (CE) in Eq.(4.17) and the Conservation of Intensity (CI) in Eq.(4.16) with six different regularization terms will produce 12 different algorithms in potential formulation and 10 different algorithms in u-v formulation. Note that there is no regularization term R_1 in u-v formulation. In addition, the combination of the regularization terms $R_1 + R_2$ and $R_1 + R_3$ which ensure the uniqueness of the solution in stream function formulation and $R_2 + R_3$ in u-v formulation with both data terms produce another six different algorithms. The values in the table represent the mean angular error of reconstructed vector fields for the Hyperbolic Fig. 2.3 data set from all 28 different combinations of algorithms. MAE from the stream function formulation are represented in Second and Third Columns whereas the MAE from u-v formulation are represented in fourth and fifth columns. The regularization term $R_\square$ in the last row of the table represents R_1 in stream function formulation and R_2 in u-v formulation. When the formulation does not exist, the MAE is replaced by an $*$.

Regularization	Stream/Potential		u-v Method	
Terms	CE	CI	CE	CI
R_1	0.096°	0.062°	$*$	$*$
R_2	0.091°	0.062°	1.706°	1.141°
R_3	0.082°	0.057°	47.094°	27.383°
R_4	0.093°	0.062°	2.084°	1.509°
R_5	0.100°	0.062°	1.400°	1.509°
R_6	0.093°	0.062°	2.292°	1.576°
$R_1 + R_2$	0.091°	0.062°	$*$	$*$
$R_\square + R_3$	0.088°	0.057°	1.706°	0.602°

Table 4.2 shows the mean angular error for the 16 different stream function formulations and 12 different u-v formulation for the hyperbolic data set. The stream function formulation with all regularization schemes produces better results than the u-v formulation. When we compare the results with respect to the data terms, the data term with conservation of energy performs well relative to the data term with continuity equation in both the stream function and the u-v formulation. The best

results are obtained from the stream function formulation with conservation of the intensity data term and the regularization term R_3 with 0.057° mean angular error.

Table 4.3: The table provides the MAE of reconstructed vector fields for the Gyre data set in Fig. 2.4 from all 28 different combinations of algorithms. MAE from the stream function formulation is represented in Second and Third Columns whereas the MAE from u-v formulation is represented in the fourth and fifth columns. The regularization term $R_\square$ in the last row of the table represents R_1 in stream function formulation and R_2 in u-v formulation. When the formulation does not exist, the MAE is replaced by an $*$.

Regularization Terms	Stream/Potential		u-v Method	
	CE	CI	CE	CI
R_1	20.749°	0.062°	$*$	$*$
R_2	3.195°	1.100°	42.680°	2.624°
R_3	9.813°	1.637°	40.991°	48.573°
R_4	10.206°	1.746°	43.133°	2.610°
R_5	9.843°	1.713°	41.975°	2.618°
R_6	10.751°	1.721°	36.376°	2.661°
$R_1 + R_2$	3.897°	1.100°	$*$	$*$
$R_\square + R_3$	32.064°	21.863°	40.560°	2.489°

Table 4.3 displays the computed MAE for the 16 different stream function formulations and 12 different u-v formulations for the Gyre data set. Most of the time, the stream function formulations with both data fidelities produce better results than the u-v formulations. By comparing the results with respect to the data terms, the data term with conservation of energy performs well relative to the data term with the continuity equation in both stream function and the u-v formulations. The best MAE is obtained as 0.057° from the stream function formulation with conservation of intensity data term and the regularization term R_1. The conservation of energy data term performs well in both stream function and u-v formulations, but the results show that there is a considerable difference in capturing the flow between R_3 using stream function formulation versus the u-v formulation.

Table 4.4: The mean angular error for the reconstructed vector fields for the source data set in Fig. 2.5 from all 28 different combinations of algorithms. MAEs from the stream function formulation represent in Second and Third Columns whereas the MAEs from u-v formulation represent in fourth and fifth columns. The regularization term $R_\square$ in the last row of the table represents R_1 in stream function formulation and R_2 in u-v formulation. When the formulation does not exist, the MAE is replaced by an $*$.

Regularization Terms	Stream/Potential		u-v Method	
	CE	CI	CE	CI
R_1	8.622°	2.755°	$*$	$*$
R_2	1.616°	2.756°	2.710°	0.647°
R_3	11.560°	1.275°	8.678°	1.104°
R_4	4.110°	2.170°	2.782°	0.895°
R_5	4.028°	2.144°	2.666°	0.895°
R_6	6.002°	2.170°	0.957°	1.228°
$R_1 + R_2$	1.616°	2.169°	$*$	$*$
$R_\square + R_3$	8.622°	0.782°	2.700°	0.647°

Table 4.4 shows the MAE for the 16 different stream function formulations and 12 different u-v formulations for the source data set. Unlike the Hyperbolic data and the Gyre data, u-v formulations with all regularization schemes produce better results than the stream function formulation. When we compare the results with respect to the data terms, the data term with conservation of energy performs well relative to the data term with the continuity equation in both stream function and the u-v formulations. However, in the stream function formulation with the regularization terms R_2 and $R_1 + R_2$ and u-v formulation with R_6, the data term with the continuity equation performs well. The best result is obtained from the u-v formulation with conservation of intensity data term and the regularization term $R_2 + R_3$ with 0.647° mean angular error.

4.6 Regularization Parameter Selection

As we have seen throughout this chapter, regularization plays an important role in solving inverse problems. The data fidelity in the optical flow problem mainly depends on data which consists of errors from measurements and discretization. Including a regularization term helps to dampen the errors and reach the desired solution. However, the amount of the regularity we emphasize in the energy functional depends on the data, the data term and the regularization term and hence it can affect to the solution. If we add too much regularity to the functional, then the regularized solution would divert from the solution we were looking for. On the other hand, if we do not add enough regularity, the solution will minimize the residuals in the original problem, but the accumulated errors will not be damped. This implies that the regularization parameter controls the quality of the regularized solution. Therefore, the best regularization parameter balances the regularization error and the errors in the residuals of the regularized solution. The selection of the best regularization parameter in optical flow problems is as important as all the other applications of inverse problems. However, the techniques for the selection of the best regularization parameter in the optical flow problems are not often available in the literature and is still an open problem with great interest.

We employ general techniques of selection of appropriate regularization parameter in the applications of inverse problems. In this area, the most common methods are L-curve [13, 69, 70], U-curve [71, 72], or generalized cross-validation (GCV) and its generalizations [73,74]. However, we have to be careful when we apply these methods as they have their own limitations [75–77]. First we will discuss these three methods briefly applying to the functional in Eq.(4.12) with the Tikhonov regularization term.

L-curve Method

In the *L-curve* method, the regularization parameter is selected based on a graph of the norm of the regularized solution versus the norm of the corresponding residuals [70] for a set of all possible regularization parameter values. If u_α is the regularized solution corresponding to the parameter α, then the norm of the residuals and the norm of the solution may be written as

$$\begin{aligned} x(\alpha) &= \|Au_\alpha - z\|_2^2 \text{ and} \\ y(\alpha) &= \|u_\alpha\|_2^2. \end{aligned}$$

For a set of positive α values, L-curve is a *log-log* scale of the parametric curve $(x(\alpha), y(\alpha))$. This curve is called the L curve as its shape is like the letter L. The L curve consists of vertical and almost horizontal parts. The vertical part of the curve represents the places where the regularization parameter is small whereas the horizontal part of the curve represents large values of the regularization parameter. According to the way that the regularization parameter varies, the vertical part of the curve is dominated by the residuals and the horizontal part is dominated by the regularization term. Therefore, in general the solutions corresponding to the vertical part are not smooth enough while the horizontal part are too smooth. This tells us a natural way to pick the best regularization parameter is by selecting the best regularization parameter, which is to select the corner point of the L-curve.

The next step is to find the corner point of the L-curve. The corner point of the L-curve is defined in two ways [15, 78]. Either as the closest point to the origin or as the maximum curvature on the curve. When it is computed numerically, the latter definition is more often used. We can consider the L-curve is a parametric curve of

$$(\hat{\rho}/2, \hat{\eta}/2) = (\log \|Au_\alpha - z\|_2, \log \|u_\alpha\|_2),$$

where $\hat{\rho}$ and $\hat{\eta}$ are functions of α. The curvature κ of the L-curve is obtained from the following relationship as in [78]

$$\kappa = 2\frac{\hat{\rho}^{'}\hat{\eta}^{''} - \hat{\rho}^{''}\hat{\eta}^{'}}{((\hat{\rho}^{'})^2 + (\hat{\eta}^{'})^2)^{3/2}}, \tag{4.30}$$

where $\hat{\rho}^{'}$, $\hat{\eta}^{'}$, $\hat{\rho}^{''}$ and $\hat{\eta}^{''}$ are the first and second derivatives of $\hat{\rho}$ and $\hat{\eta}$ with respect to α respectively. The best regularization parameter is the α where the curvature κ is maximized. Even though the L-curve criteria is discussed for the problem in Eq.(4.12) which involves the Tikhonov regularization term, we can extend the criteria as in [78, 79] to a general regularization term

$$\arg\min_{u} \|Au - z\|_2^2 + \alpha\|Lu\|_2^2,$$

where L is the regularization operator. In the general case, we can define $x(\alpha)$ and $y(\alpha)$ as

$$\begin{aligned} x(\alpha) &= \|Au_\alpha - z\|_2^2 \\ y(\alpha) &= \|Lu_\alpha\|_2^2 \end{aligned} \tag{4.31}$$

and the rest is the same as the procedure of the problem with Tikhonov regularization.

U-curve Method

In the U-curve method, we consider the sum of the reciprocals of the solution norm and the residual norm against the regularization parameter α. This can be represented as [15]

$$U(\alpha) = \frac{1}{x(\alpha)} + \frac{1}{y(\alpha)}, \tag{4.32}$$

where $x(\alpha)$ and $y(\alpha)$ are defined as in Eq.(4.31). Similar to the L curve, since the shape of this curve is close to the letter 'U', the method is called the U-curve method.

In the U-curve there are two almost vertical parts in both the left and right sides and an almost horizontal part between those two vertical parts.The two vertical parts represent the regularization parameter values which either the solution norm or the residual norm dominates over each other and the horizontal part represents the α values where the solution norm and the residual norm are close to each other. The best regularization solution from the U-curve method is obtained at the minimum of $U(\alpha)$. In [15], it is proven that the function $U(\alpha)$ has a local minimum in the interval $\alpha \in \left(\sigma_1^{2/3}, \sigma_r^{2/3}\right)$, where σ_1 is the largest singular value of the operator A, and σ_r is the smallest non-zero singular value of the operator A.

Genaralized Cross Validation Method

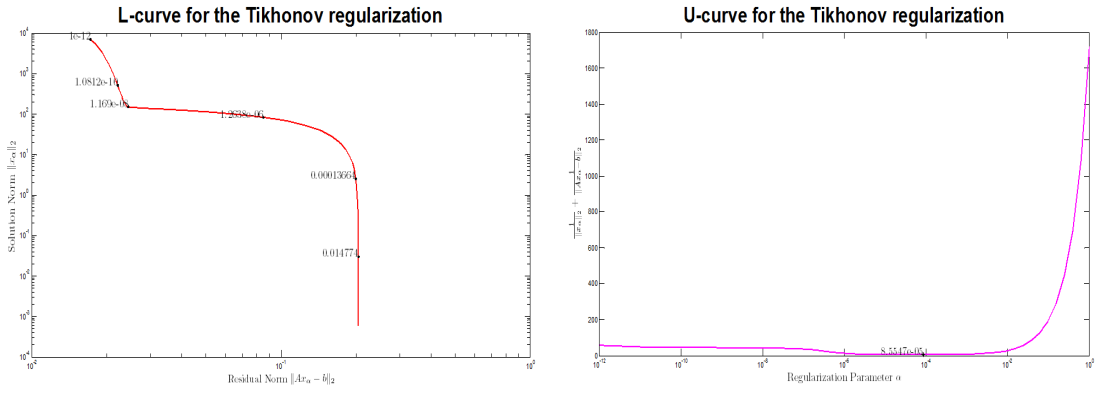
The third method we use to compute the best regularization parameter is the Generalized Cross Validation method. In this method, we consider what happens if we remove one data point in the image. We then search for the best parameter that predicts the deleted data point. We obtain this optimal α by minimizing the function

$$G(\alpha) = \frac{\frac{1}{n}\|Au_\alpha - z\|_2^2}{\left[\frac{1}{n} trace\left(I - A\left(A^TA + \alpha L\right)^{-1} A^T\right)\right]^2}, \tag{4.33}$$

where n is the length of the vector z.

To demonstrate the L-curve and U-curve methods, we apply stream function formulation with conservation of energy data fidelity and the Tikhonov regularization term on each of the gyre data and the hyperbolic data.

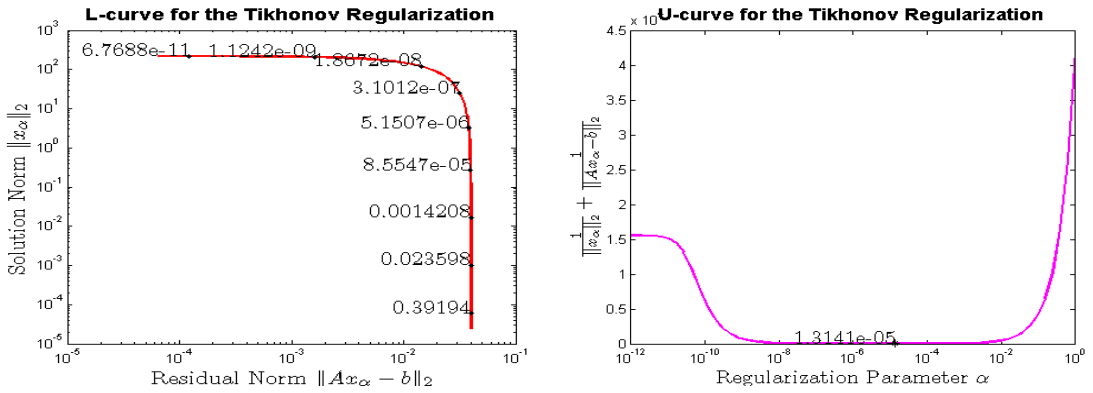
Fig. 4.5 shows L-curve and the U-curve plots for the gyre data using conservation of intensity data term and the Tikhonov regularization term. The suggested regularization values from L-curve method is 1.169×10^{-8} and U-curve solution is 8.555×10^{-5}. The Mean angular values corresponding to the α values suggested



(a) (b)

Figure 4.5: L-curve and U-curve on gyre data – The images (a) and (b) show the L-curve and U-curve plot plots for the stream function approach with the conservation of energy data fidelity and the Tikhonov regularization term on the gyre data set.

from L-curve and the U-curve methods are 48.930° and 47.077° respectively. The minimum mean angular error that can be obtain is 44.974° and the corresponding $\alpha = 6.261 \times 10^{-11}$. In this case, both L-curve and the U-curve solutions are not in a neighborhood of best α However, the Tikhonov regularization is not capable of capturing the gyre flow as the minimum MAE is approximately 45°.



(a) (b)

Figure 4.6: L-curve and U-curve on hyperbolic data – The image (a) shows the L-curve plot for the stream function approach with the conservation of energy data fidelity and the Tikhonov regularization term on the hyperbolic data set. The image (b) shows the U-curve plot from the same formulation on the hyperbolic data set.

Images (a) and (b) in Fig. 4.6 shows the L-curve and the U-curve plots for the hyperbolic data set using conservation of intensity and the Tikhonov regularization term. In this case, the L-curve method is unable to produce the L shape rather than compute the best regularization parameter α. The U-curve method suggests $\alpha = 1.314 \times 10^{-5}$ and the associate MAE is 10.741°. However, when $\alpha = 1.536 \times 10^{-13}$, the same algorithm reconstructs the hyperbolic flow with the MAE is 0.658° which is the minimum.

The L-curve and the U-curve methods do not provide reasonable solutions to the above two examples. Therefore, to further test the determination of the best regularization parameter α, we applied L-curve method, U-curve method and the GCV method on the gyre data set shown in Fig. 2.4. In this demonstration, the stream function formulation with conservation of intensity data fidelity is employed by with the regularization terms R_2 and R_4 one at a time. Each method provides the best regularization parameters according their own abilities. The image (a) in Fig. 4.7 shows a graph of mean angular error versus regularization parameter for each formulation. In stream function formulations with R_4, the U-curve method determines a suitable regularization parameter value which is close to the minimum mean angular error. The outcomes from the L-curve and GCV methods leads to higher mean angular errors. However, in the second approach with the regularization term R_2, none of the three methods provides a suitable parameters which close to the minimum mean angular error. Hence the generalization of these methods are not appropriate and selection of the regularization parameter remains as an important future work.

When we presented the results in this Chapter for the synthetic flows, we selected the best regularization parameter by considering the minimum mean angular error. For this purpose, we consider 100 equally spaced parameter values from 10^{-14} to 10^{0} and compute mean angular error for each parameter value. Then the best alpha will

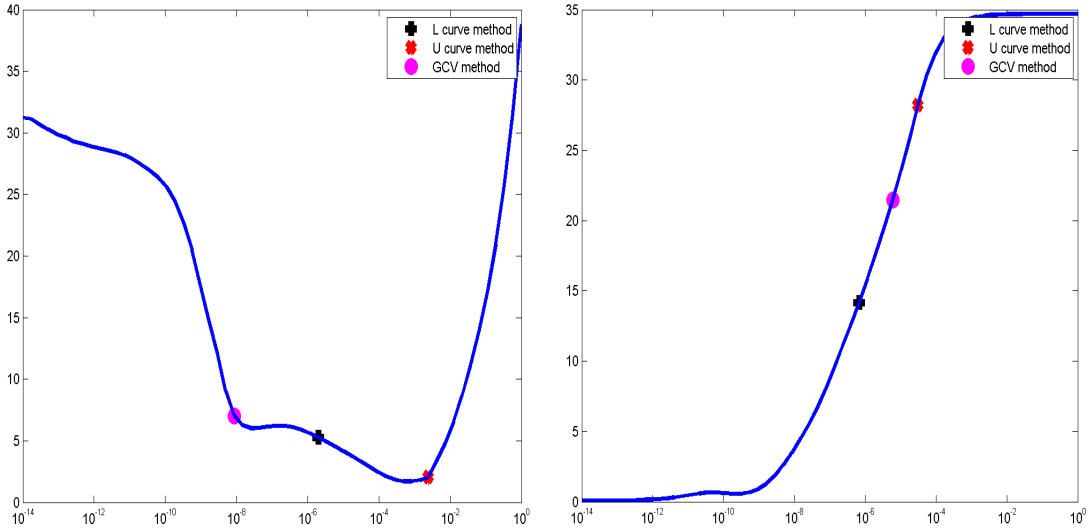


Figure 4.7: The best regularization parameter – The image (a) represents the graphs of mean angular error versus regularization parameter from the stream function method with conservation of Intensity data fidelity and the regularization term R_4 on the gyre data set. The image(b) is also a graph mean angular error versus regularization parameter with the same data set and the same formulation replacing R_4 by R_2. In each case, the best regularization parameters determined by the L-curve, U-curve, and GCV methods are highlighted.

provide the minimum mean angular error.

In real world applications, there is no way to compute the MAE. Therefore, selection of the best solution plays an important role in reconstructing vector fields. In this case, we use a basic approach to select the regularization parameter by considering the relative changes in velocity components corresponding to different regularization parameter values. Also in this approach is also, we consider 100 equally spaced parameter values from 10^{-14} to 10^{0} and compute the velocity fields for the each parameter value. Then the norm of the error for each consecutive parameter is computed, and the best regularization parameter is selected from the minimum error.

Chapter 5

Lagged Diffusivity Fixed Point Iteration Method

In earlier chapters, we presented various algorithms for computing optical flow for both rigid body as well as fluid motion. First we introduced the original optical flow computation approach developed by Horn and Schunck in [9] which assumes image brightness is conserved locally and that the expected flow field is smooth. Recall that the corresponding energy functional is

$$E(u,v) = \int_\Omega \left(I_t + I_x u + I_y v\right)^2 \, d\Omega + \alpha \int_\Omega \left(u_x^2 + u_y^2 + v_x^2 + v_y^2\right) \, d\Omega. \tag{5.1}$$

The later researchers extended this energy functional for special cases by introducing new data fidelities [29, 40] and regularization terms [41, 80]. Further some researchers [42] developed numerical methods to enhance both the accuracy and the convergence of the optical flow algorithms. In addition to these natural extensions, some researchers [37,38,81] introduced new optical flow algorithms for fluid motion by computing governing stream function between two time instances rather than computing velocity components. In this case, the velocity components are determined from the estimated stream/potential function.

In this chapter, we consider another regularization term, the Total Variation (TV) regularization term and then we adopt for the first time the Lagged Diffusivity Fixed Point Method [49] to solve the optical flow algorithm [55]. The TV regularization term was originally introduced in [82] with the purpose of removing noise from images. Thereafter this regularization term has been activated for image reconstructions [83–85] from noise and blurred images as it preserves the edges of the images. Later, the TV regularization term started to appear in optical flow algorithms [27, 86, 87] to compute flow fields with discontinuities. For instance, we can expect discontinuities in the turbulent structures found in fluid motion.

The total variation for a one-dimensional real-valued function f on the interval $[a, b]$ can be defined as

$$TV\left(f\right) = \sup_{\mathcal{P}} \sum_{i=1}^{n} |f\left(x_i\right) - f\left(x_{i-1}\right)|, \tag{5.2}$$

where $\mathcal{P}$ is a partition of $[a, b]$ so that $a = x_0 < x_1 < x_2 \ldots < x_n = b$. This term is able to measure the jump discontinuities of the function f. For example, if f is a piecewise constant function, then the total variation of the function f on $[a, b]$ is equal to the sum over the magnitudes of jumps for finite jumps. For a smooth function f, the Eq.(5.2) can be written as

$$TV\left(f\right) = \sup_{\mathcal{P}} \sum_{i=1}^{n} \left| \frac{f\left(x_i\right) - f\left(x_{i-1}\right)}{\Delta x} \right| \Delta x, \tag{5.3}$$

where $\Delta x = x_i - x_{i-1}$. When the $\Delta x \to 0$, the continuous formulation is given as

$$TV\left(f\right) = \int_{a}^{b} |f'(x)| \, dx. \tag{5.4}$$

We can extend the definition for a function f of two variables x and y on the domain Ω as

$$TV\,(f) = \int_{\Omega} |\nabla f|\; d\Omega. \tag{5.5}$$

When we use the TV regularization term, it measures the jump discontinuities allowing us to penalize the piecewise fluctuations of the regularized solution. The inclusion of TV regularization in optical flow algorithms in u-v formulation may result in a piecewise constant velocity field. However, the optical flow algorithm with TV regularization in the stream function formulation may lead to a piecewise constant stream function and hence to a sparse flow. First, we discuss the flow computation with TV regularization in u-v formulation and then in the stream function formulation.

5.1 Optical flow with Total Variation Regularization

For the flows with some discontinuities, we can use the total variation of the flow components u and v as the regularization term. The energy functional with the conservation of intensity (CI) data term can be written as

$$E(u,v) = \int_{\Omega} \left(I_t + I_x u + I_y v\right)^2\; d\Omega + \alpha \int_{\Omega} \left(|\nabla u| + |\nabla v|\right)\; d\Omega. \tag{5.6}$$

Now applying Eq.(2.19) on the functional in Eq.(5.6), we have the Euler-Lagrange equations for Eq.(5.6) as

$$\begin{aligned} I_x(I_t + I_x u + I_y v) + \alpha \nabla \cdot \left(\frac{\nabla u}{|\nabla u|}\right) &= 0 \\ I_y(I_t + I_x u + I_y v) + \alpha \nabla \cdot \left(\frac{\nabla v}{|\nabla v|}\right) &= 0. \end{aligned} \tag{5.7}$$

Note that here we replaced α by 2α for a convenient simplification. Now neither of the Euler-Lagrange equations are linear as the diffusion coefficients $1/|\nabla u|$ and $1/|\nabla v|$ are present in the equations. Due to the difficulty of linearization, linear numerical solution methods such as the LU-factorization method are not appropriate for solving the system. We can, however, use gradient-based methods to solve the system. First we apply the gradient descent method for solving the Euler-Lagrange equations in Eq.(5.7) by updating the solution via an iterative procedure. Before we apply the gradient descent method, we rewrite $|\nabla u|$ and $|\nabla v|$ as $\sqrt{|\nabla u|^2+\epsilon}$ and $\sqrt{|\nabla v|^2+\epsilon}$ respectively [27, 49], where ϵ is a small real number. This step is required to determine the divergence of $\frac{\nabla u}{|\nabla u|}$ in Euler-Lagrange equations as it enforces the differentiability at the origin. This also helps to avoid the zeros in the denominator, when we numerically solve it. Then the gradient descent approach will be

$$\begin{aligned} u^{n+1} &= u^n + \Delta\tau \left[I_x(I_t + I_x u + I_y v) + \alpha \nabla \cdot \left(\frac{\nabla u}{\sqrt{|\nabla u|^2+\epsilon}} \right) \right] \\ v^{n+1} &= v^n + \Delta\tau \left[I_y(I_t + I_x u + I_y v) + \alpha \nabla \cdot \left(\frac{\nabla v}{\sqrt{|\nabla v|^2+\epsilon}} \right) \right], \end{aligned} \tag{5.8}$$

where n and τ are the iteration number and the step size for each iteration respectively. The partial derivatives of u, v and I are approximated using the finite difference approximations.

In addition to the conservation of intensity data term, we can use the continuity equation (CE) data term with the TV regularization term to compute the optical flow. The corresponding energy functional is written as

$$E(u,v) = \int_\Omega \left(I_t + I_x u + I_y v + I u_x + I v_y\right)^2 \, d\Omega + \alpha \int_\Omega \left(|\nabla u| + |\nabla v|\right) \, d\Omega. \tag{5.9}$$

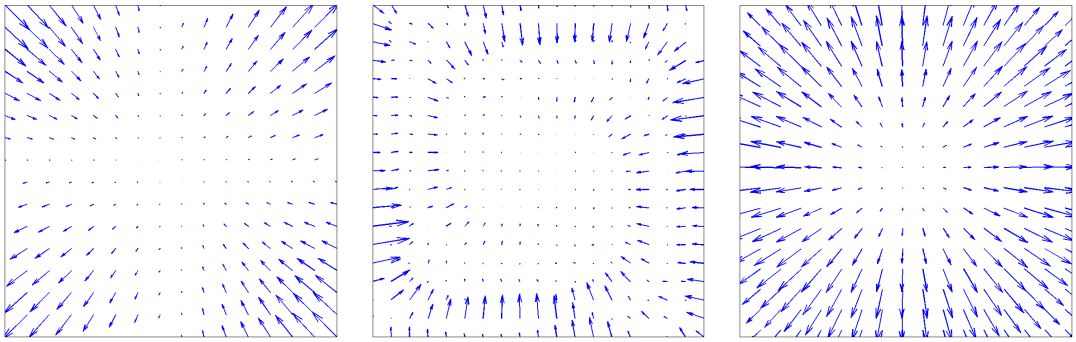
The Euler-Lagrange equations for Eq.(5.6) are obtained by applying Eq.(2.19) on the functional in Eq.(5.6) and written as

$$\begin{aligned} I(I_{tx} + I_{xx}u + 2I_x u_x + I_{yx}v + I_y v_x + Iu_{xx} + I_x v_y + Iv_{yx}) + \alpha\nabla \cdot \left(\frac{\nabla u}{|\nabla u|}\right) = 0 \\ I(I_{ty} + I_{xy}u + I_x u_y + I_{yy}v + 2I_y v_y + Iu_{xy} + I_y u_y + Iv_{yy}) + \alpha\nabla \cdot \left(\frac{\nabla v}{|\nabla v|}\right) = 0. \end{aligned} \tag{5.10}$$

To solve the Euler-Lagrange equations in Eq.(5.10), we use the gradient descent approach as explained in Eq.(5.8) for the equations in Eq.(5.7). Now we apply both algorithms to compute the optical flow on the three data sets, hyperbolic fixed point, gyre and source data which we introduced in Fig. 2.3, Fig. 2.4 and Fig. 2.5 respectively.

Fig. 5.1 shows the computed flow from the conservation of intensity (CI) data term and the TV regularization term in u-v formulation on the three synthetic data sets. The flow fields are computed using a gradient descent approach. The resulting flow fields for the hyperbolic fixed point, gyre and the source data are shown in images (a), (b), and (c) respectively. The algorithm is capable of reconstructing the source flow accurately whereas it is unable to capture even the structure of the gyre. In the hyperbolic flow, the diagonal flows are accurate, but the center is not. The computed mean angular errors (MAE) for the hyperbolic, gyre and source flows are presented accordingly in the second column of the Table 5.1. It can be observed that only the reconstructed source flow has a reasonable MAE.

The reconstructed flow fields from the continuity equation data term with the TV regularizer in u-v formulation are shown in Fig. 5.2. Images (a), (b), and (c) represent the reconstructed hyperbolic, gyre and source flows respectively. We used the gradient descent method as the numerical scheme to solve the Euler-Lagrange equations. The reconstructed hyperbolic flow is reasonable as the MAE is 2.759° as shown in the third column of the Table 5.1. The gyre flow has a high MAE and also

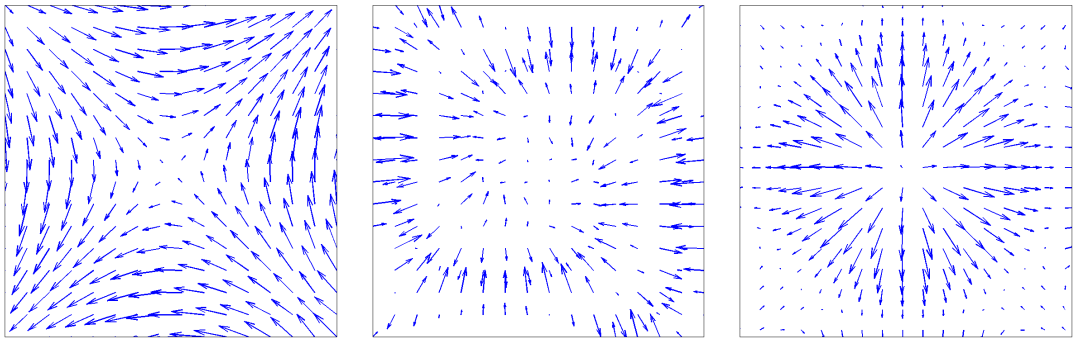


(a) (b) (c)

Figure 5.1: Flow from CI+TV with gradient descent method – Reconstructed flow fields for the hyperbolic, gyre and source data from the u-v formulation with the conservation of intensity data term and the TV regularization term. The source flow is very close to the true flow as the MAE is 1.650°, but the others are not reasonable.

it is observed that the reconstruction is not even close to the true flow field. The computed source flow captured the center very accurately, but the points near the boundaries accompanied errors.

Mean angular errors for both the conservation of intensity and the continuity equation data fidelities with TV regularization term are shown in Table 5.1. Here we use the u-v approach with gradient descent method as the numerical scheme to solve the Euler-Lagrange equations. The second column shows the MAEs for the algorithm with conservation of intensity data fidelity, and the third column represents the errors for the algorithm with continuity equation data term. Mean angular errors for the hyperbolic, gyre and source flows are shown in second, third and fourth rows respectively. When we consider the results in Chapter 4, the best reconstructions for the hyperbolic, gyre and source flows are obtained from CI+R_3 in stream function formulation, CI+R_1 in stream function formulation and CI+R_2 in u-v formulation with the MAEs 0.057°, 0.062°, and 0.647° respectively. According to these results, all three flows structures should be able to reconstruct the desired solutions. When we consider both TV approaches, at least one algorithm reconstructs hyperbolic and



(a) (b) (c)

Figure 5.2: Flow from CE+TV with gradient descent method – Images (a), (b), and (c) show the reconstructed flow fields for the hyperbolic, gyre and source data from the TV regularization and continuity equation data fidelity in u-v formulation. The MAE for the hyperbolic flow is 2.759°, whereas other two flows have high MAEs. As we can see, the hyperbolic flow is reasonable and the source flow is accurate at the center but not near the boundaries. Also the algorithm is unable to reconstruct the gyre flow.

source flow accurately, but none of them are capable of reconstructing the gyre flow.

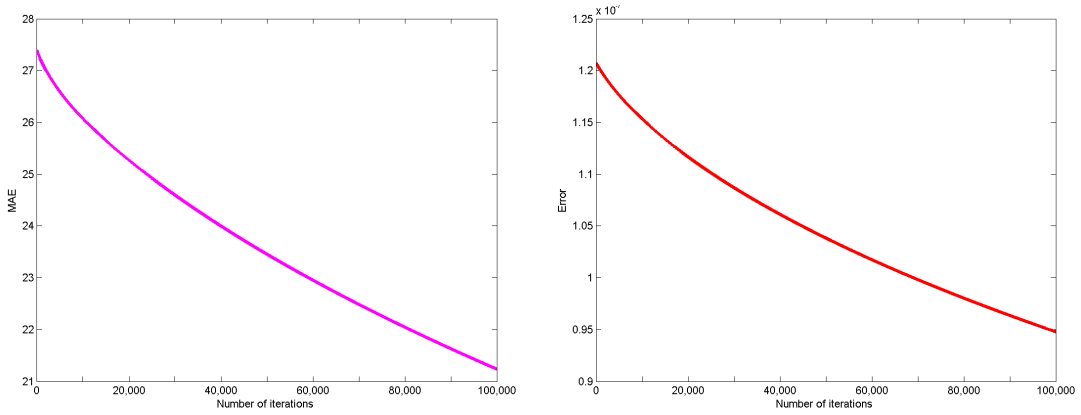
For the stopping criteria of the gradient descent method, we set $n_{max} = 100,000$ and the threshold value $\delta = 10^{-10}$. When either the number of iterations reaches n_{max} or the error of the flow differencing between successive iterates is smaller than δ. Images (a) and (b) in Fig. 5.3 show the mean angular error verses the number of iterations plotted and the errors of successive flows verses number of iterations

Table 5.1: Mean Angular Errors for the hyperbolic fixed point, gyre, and source flows using TV regularizer with conservation of intensity (CI) and continuity equation (CE) data fidelities are shown in the second and third columns respectively. In this case, the u-v formulation is used with the gradient descent method.

Flow/Method	CI+TV in u-v	CE+TV in u-v
Hyperbolic	21.232°	2.759°
Gyre	47.174°	58.871°
Source	1.650°	26.058°

respectively. Due to the time constraint, we stopped the algorithms after 100,000 iterations even though both errors were improving. Note that the flow errors of successive iterates is smaller than 10^{-7} in many iterations, but the mean angular error of 21.232° is relatively high even after 100,000 iterations. This tells us that the algorithm converges slowly.

For each of these computations, the regularization parameters α and ϵ must be selected. In these computations, we have performed an exhaustive search to select α so that it minimizes the mean angular error. The regularization parameters chosen for the hyperbolic, gyre and source data sets with the conservation of intensity data term are $\alpha = 0.122$, 0.93 and 8.92×10^{-2}, respectively. The parameter values corresponding to the continuity equation data fidelity are $\alpha = 10^{-4}$, 1.21×10^{-3} and 3.16×10^{-4}, respectively. In each case, the minimum mean angular error corresponded to $\epsilon = 10^{-2}$.



(a) Mean angular error

(b) Successive iterates error

Figure 5.3: Computed mean angular error and the flow errors of successive iterates for 100,000 iterations on hyperbolic flow are shown in images (a) and (b) respectively. We applied the gradient descent approach with the conservation of intensity data term and the TV regularizer in u-v formulation to compute the errors. Both of the errors are improving even after 100,000 iterations.

The main disadvantage of the gradient descent approach is that the convergence is slow. All the above results in Fig. 5.1 and Fig. 5.2 are obtained after 100,000

iterations. Also, the results from both the conservation of intensity and the continuity equation data fidelities are not quantitatively comparable with the results we obtained in Chapter 4 with respect to the mean angular errors. Due to these facts, we modified the algorithm by introducing an existing numerical method to solve the system of Euler-Lagrange equations in Eq.(5.7). The new numerical approach is the Lagged Diffusivity Fixed Point Method [49] which was used in [88] to solve a total variation based image denoising problem.

5.2 Lagged Diffusivity Fixed Point Iteration (LDFPI) Method

The slow convergence of gradient descent method on the TV regularized optical flow algorithms motivated us to adopt the Lagged Diffusivity Fixed Point Iteration Method in optical flow algorithms. First we present the Lagged Diffusivity Fixed Point algorithm as explained in [49] to determine the solution of the operator equation $kf = g$ from the total variation regularized functional

$$E(f) = \|Kf - g\|_2^2 + \alpha TV(f), \tag{5.11}$$

where f is the function to be determined, K is a matrix and g is the measured data. As the preliminary step, we use the standard approximation to the total variation of f as

$$TV(f) = \int_\Omega |\nabla f| \, d\Omega \approx \int_\Omega \sqrt{f_x^2 + f_y^2 + \epsilon} \, d\Omega,$$

where ϵ is a small positive real number which is fixed. We do this step to make sure the differentiability of the TV term at the origin as the Euler-Lagrange equation of the TV term involves computing partial derivatives. For a given initial condition f_0

and letting $P_n = \frac{1}{\sqrt{f_{n_x}^2 + f_{n_y}^2 + \epsilon}}$, the rest of the algorithm can be expressed in five steps as listed below.

1. Discretize the regularizer as an operator: $L_n = D_x^T P_n D_x + D_y^T P_n D_y$, where D_x and D_y are partial derivative operator matrices with respect to x and y
2. Determine the gradient of E: $G_n = K^T(Kf_n - g) + \alpha L_n f_n$
3. Approximate the Hessian matrix: $H = K^T K + \alpha L_n$
4. Quasi-Newton step: $s_n = -H^{-1} G_n$
5. Update the approximate solution: $f_{n+1} = f_n + s_n$

We can apply the above LDFPI method to solve the TV regularized optical flow problem.

5.2.1 Optical Flow with LDFPI method

We now apply the Lagged Diffusivity Fixed Point Iteration method to our optical flow computation. First, we consider the optical flow functional in Eq.(5.6) and solve the corresponding Euler-Lagrange equations in Eq.(5.7). In this application, $f = \langle u, v \rangle$, $K = [-I_x, -I_y]$, and $g = I_t$. Considering the flow components u and v as column vectors, as stated above, we define

$$P_u = \frac{1}{\sqrt{u_x^2 + u_y^2 + \epsilon}} \quad \text{and} \quad P_v = \frac{1}{\sqrt{v_x^2 + v_y^2 + \epsilon}}.$$

Using this definition, we can discretize the gradients of the $TV(u)$ and $TV(v)$ as

$$\begin{aligned} \nabla . \left(\frac{\nabla u}{|\nabla u|} \right) &= (D_x^T P_u D_x + D_y^T P_u D_y) u \\ \nabla . \left(\frac{\nabla v}{|\nabla v|} \right) &= (D_x^T P_v D_x + D_y^T P_v D_y) v. \end{aligned} \tag{5.12}$$

The discretized operator for the Euler-Lagrange equations of $TV(u) + TV(v)$ can be written as

$$L_n = \begin{bmatrix} D_x^T P_{u_n} D_x + D_y^T P_{u_n} D_y & 0 \\ 0 & D_x^T P_{v_n} D_x + D_y^T P_{v_n} D_y \end{bmatrix},$$

where n is the iteration number. Since we have already discretized the operator L_n, we need to follow only the last four steps of the LDFPI algorithm. The four steps are given as

1. Compute gradient direction: $g_n = K^T(K[u_n, v_n]^T - g) + \alpha L_n[u_n, v_n]^T$
2. Approximate Hessian: $H_n = K^T K + \alpha L_n$
3. Quasi-Newton Step: $H_n[w_n, y_n]^T = -g_n$
4. Solution update: $[u_{n+1}, v_{n+1}] = [u_n, v_n] + [w_n, y_n]$

We can follow the same procedure for the optical flow energy functional with the continuity equation data fidelity and the TV regularization in Eq.(5.9) replacing

$$K(u, v) = -I_x u - I_y v \quad \text{by} \quad K(u, v) = -I_x u - I_y v - I u_x - I v_y.$$

For both algorithms, we need to find a suitable initial condition $[u_0^T, v_0^T]$ and an optimal iteration number n^*. We chose zero vectors for the initial vector fields in these algorithms. To select a reasonable n, as we did for the gradient descent method, we first measure the errors between the flow of the preceding step and the current step. We then run the algorithm until we meet the condition that the error is smaller than a given threshold or the n reaches n_{max}. When the algorithm reaches one of these stopping criteria, we stop the iterative procedure. For LDFPI method, the maximum

number of iterations is $n_{max} = 100$ and the threshold value is 10^{-10}. The other important step of this algorithm is the Quasi-Newton step in the LDFPI method which involves numerical calculations determining the inverse of a large matrix. In this case, we use an LU factorization on H_n at each iteration and then Gaussian elimination to determine the improvements $[w_n, y_n]^T$.

We applied the conservation of intensity data term with the TV regularization algorithm on the three synthetic data sets: hyperbolic fixed point, gyre and source flow. Images (a), (b) and (c) in Fig. 5.4 show the computed flow fields on hyperbolic fixed point, gyre and the source flow respectively. All three reconstructions are qualitatively very close to the true solutions. Also the mean angular errors as shown in Table 5.2 corresponding to all three constructions are below 2.6° and shows the quantitative accuracy of this method.

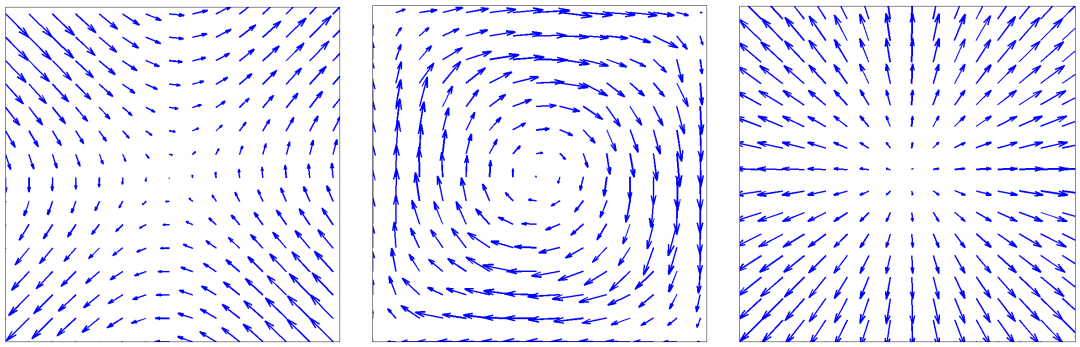
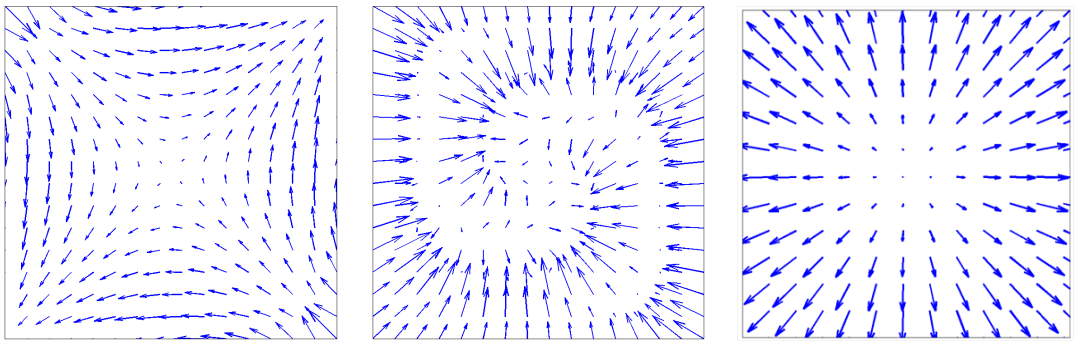


Figure 5.4: Computed flow from CI data fidelity– Images (a), (b) and (c) show the computed velocity fields from the conservation of intensity data fidelity with TV regularization term for the saddle, gyre and source flow images shown in Fig. 2.3, Fig. 2.4 and Fig. 2.5 respectively. All the reconstructions are very accurate as they have smaller mean angular errors as shown in Table 5.2.

Images (a), (b) and (c) in Fig. 5.5 shows the computed flow fields from the continuity equation data fidelity and the TV regularization term on the three data sets, hyperbolic fixed point in Fig. 2.3, gyre in Fig. 2.4 and the source flow in Fig. 2.5

respectively. The algorithm captures the source flow accurately and the hyperbolic flow accurately except for a few boundary points. The algorithm is not only unable to reconstruct the gyre flow but as we have seen in our previous chapters, the algorithms with continuity equation data fidelity barely captured the gyre flow at all.



(a) (b) (c)

Figure 5.5: Computed flow from CE data fidelity– Images (a), (b) and (c) show the computed velocity fields from the continuity equation data fidelity with TV regularization term for the saddle, gyre and source images shown in Fig. 2.3, Fig. 2.4 and Fig. 2.5 respectively. Both hyperbolic and source flows are reasonable, where as the gyre flow is not even close to the true flow field.

Table 5.2 shows the mean angular errors for the hyperbolic, gyre and source flows from the LDFPI algorithm. The second column represents the MAEs from the conservation of energy data fidelity with TV regularizer and the third represents the continuity equation data fidelity with TV. The conservation of intensity based algorithm has very small mean angular errors for all three reconstructions. The algorithm with the continuity equation data term is capable of capturing the source flow very accurately as the MAE is 1.689° and of capturing the hyperbolic flow is reasonably well. However, MAE for the gyre flow is very high. This may be due to the incapability of the data fidelity as we have seen in our previous chapters. The regularization parameters chosen for the hyperbolic, gyre and Source data sets with the conservation of intensity data term are $\alpha = 10^{-14}$, 10^{-14} and 3.16×10^{-8} respectively. The parameter values corresponding to the continuity equation data

Table 5.2: Mean Angular Errors from the LDFPI method for the hyperbolic fixed point, gyre, and source flows using TV regularizer with conservation of intensity (CI) and continuity equation (CE) data fidelities are shown in the second and third columns respectively. In this case, the u-v formulation and the algorithm with the conservation of intensity data fidelity produces small MAEs.

Flow/Method	CI+TV in u-v	CE+TV in u-v
Hyperbolic	2.236°	4.772°
Gyre	2.576°	57.555°
Source	1.672°	1.689°

fidelity are $\alpha = 4 \times 10^{-5}$, 0.32 and 0.96 respectively. In this case also, the minimum mean angular error corresponded to $\epsilon = 10^{-2}$.

In general, the LDFPI algorithm with conservation of intensity data term with TV regularization term produces reasonable solutions in the u-v formulation. Our next step is to apply the LDFPI algorithm with the stream function formulation.

5.3 Lagged Diffusivity Fixed Point Method in Stream Function Formulation

We now include the TV regularization term in the stream function formulation and hence we apply the LDFPI method to solve the problem. The TV regularization term in u-v formulation tends to produce piecewise constant velocity components u and v. In stream function formulation, the TV regularization term may produce piecewise constant stream function. This yields the TV regularized solution will be favored to reconstruct sparse flow fields. Hence the u and v with TV regularization and the stream with TV regularization are not comparable. The purpose of the inclusion of the TV regularization term in the stream function formulation is that as another application of lagged diffusivity fixed point iteration method. The energy functional

of the TV regularizer with the conservation of intensity data term in the stream function formulation can be written as

$$E(\psi) = \int_\Omega \left(I_t - I_x\psi_y + I_y\psi_x\right)^2 \, d\Omega + \alpha \int_\Omega |\nabla\psi| \, d\Omega. \tag{5.13}$$

Let $K = -\left(-I_xD_y + I_yD_x\right)$ and $h = I_t$, then the energy functional in Eq.(5.13) becomes

$$E(\psi) = \int_\Omega \left(h - K\psi\right)^2 \, d\Omega + \alpha \int_\Omega |\nabla\psi| \, d\Omega, \tag{5.14}$$

and hence the Euler-Lagrange equation is

$$K^* \left(I_t - K\psi\right) + \alpha\nabla \cdot \left(\frac{\nabla\psi}{|\nabla\psi|}\right) = 0, \tag{5.15}$$

where K^* is the operator adjoint of K and we replace α by 2α. Next we discritize the gradient of the $TV(\psi)$ as

$$L = D_x^T P D_x + D_y^T P D_y, \quad \text{where} \quad P = \frac{1}{\sqrt{\psi_x^2 + \psi_y^2 + \epsilon}}.$$

Note the LDFPI algorithm for the stream function formulation for a given initial ψ_0 is given by

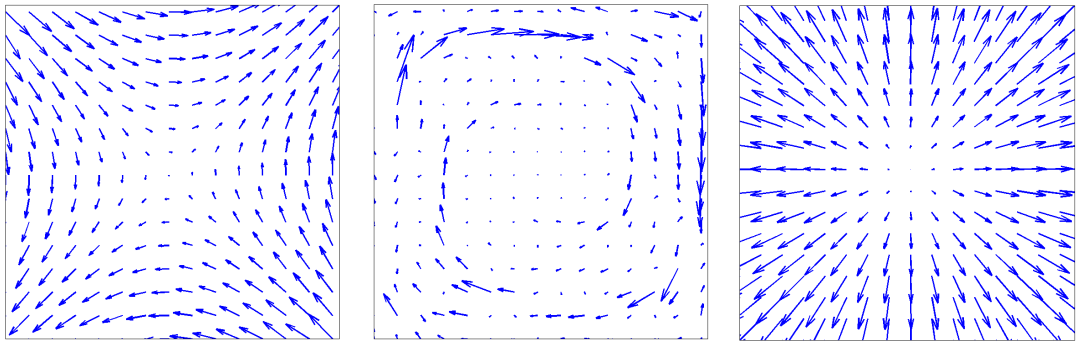
1. Compute gradient direction: $g_n = K^T(K\psi_n - h) + \alpha L_n\psi_n$
2. Approximate Hessian: $H_n = K^TK + \alpha L_n$
3. Quasi-Newton Step: $H_n\varphi_n = -g_n$
4. Solution update: $\psi_{n+1} = \psi_n + \varphi_n$,

Note that, by the definition, the gradient of the $TV(\psi)$ is equal to the gradient of the $TV(\phi)$. Therefore, the potential function formulation also follows a similar procedure

with

$$K = -I_x D_x - I_y D_y.$$

The computed flow fields from the stream formulation on the hyperbolic fixed point, gyre and source data are shown in the images (a), (b) and (c) in Fig. 5.6 respectively. The algorithm captures the hyperbolic and source flows more accurately with smaller



(a) (b) (c)

Figure 5.6: Computed flow from CI+TV stream function method – Images (a), (b) and (c) show the computed velocity fields from the stream function formulation for the hyperbolic fixed point, gyre and source images shown in Fig. 2.3, Fig. 2.4 and Fig. 2.5 respectively. Here we use the conservation of intensity data fidelity.

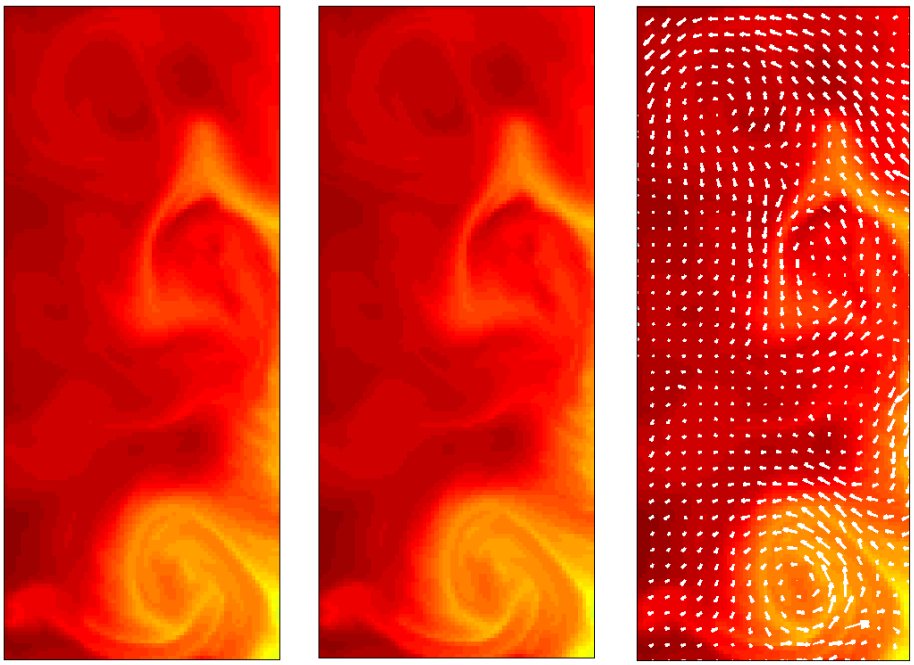
mean angular errors 0.991° and 1.45° compared to the u-v formulation with the algorithm having the same data fidelity. However, the stream formulation is not able to capture the gyre flow accurately and the corresponding mean angular error is 25.584°. In this case, the u-v formulation with the conservation of intensity data term results is an MAE of 2.576°. This is due to imposing different information on the functional when we regularize u-v and the stream formulation with TV regularizer.

The regularization parameters chosen for the hyperbolic, gyre and source data sets are $\alpha = 8.48 \times 10^{-14}$, 5.99×10^{-9} and 2.68×10^{-7} respectively for the stream formulation. Again the minimum mean angular error corresponded to $\epsilon = 10^{-2}$.

5.3.1 Flow for Oceanic Data

Next we apply an optical flow algorithm with the LDFPI method to compute the flow between time instances of the virtual flow of sea surface temperature off the coast of Oregon, U.S.A. Images (a) and (b) in Fig. 5.7 show two time-adjacent images taken one hour apart in time on August 1, 2002, representing sea surface temperature. In these images, yellow and orange regions represents warmer surface temperatures, and the red regions represents cooler surface temperatures.

According to the mean angular errors we obtained for different approaches in this



(a) (b) (c)

Figure 5.7: Sea Surface Temperature Flow Computations – Images (a) and (b) represent sea surface temperature off the coast of Oregon in August 2002 one hour apart. The lighter gray regions correspond to warmer surface temperatures and the darker regions to cooler temperatures. The computed flow with $\alpha = 10^{-5}$ is shown in (c).

chapter, the lagged diffusivity fixed point method with conservation data fidelity and the TV regularizer in u-v formulation outperforms the other approaches. Therefore, we used that approach to determine the flow fields of the sea surface temperature

data. The image (c) in Fig. 5.7 shows the reconstructed flow field. We can clearly see that the algorithm captures two vortices. The algorithm does not capture the laminar flows in between the vortices well, but it is able to capture the important structure in fluid dynamics that are of the most interest when studying the dynamics of fluid systems.

5.4 Convergence Analysis of LDFPI for Optical Flow

The main advantage of applying the LDFPI algorithm for solving TV regularized optical flow problems is that the convergence of the method is very fast irrespective of the initial condition. In general, convergence of some numerical methods depends on the initial condition, but the LDFPI methods converges fast even with zero-initial conditions.

Fig. 5.8 shows the mean angular error vs. iteration number for each of the three synthetic data sets analyzed above. Initial MAE for the zero initial flow is greater than 60°, however, in each case, MAE reduce in few iterations. Even though each iteration of the fixed point method is computationally more expensive than an iteration for an explicit-time scheme, the optical flow algorithm with the TV-regularization in u-v formulation for each of the 3 data sets converges in fewer than 10 iterations. Also the algorithm in stream function formulation for the source and hyperbolic data sets with the lagged diffusivity fixed point method converges in same number of iterations.

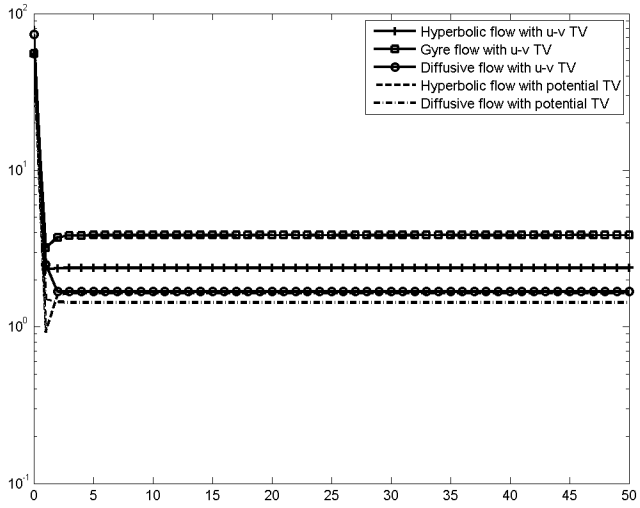


Figure 5.8: MAE vs. Iteration Number – The computed mean angular error for the first 50 iterations with a fixed α for the hyperbolic, single gyre, and source data sets.

Fig. 5.8 is a numerical demonstration of convergence of the convergence of the algorithm. The proof of the convergence of the lagged diffusivity fixed point algorithm is available in [89, 90]

Chapter 6

Multi-Time Step Method

In dynamical systems, we may have autonomous systems or non-autonomous systems. The distinction is made based on whether the velocity components are time-independent or dependent. However, autonomous systems are hardly seen in real world dynamical systems, especially not in fluid systems. Performance of most of the analytical tools in dynamical systems is based on time-dependent velocity fields. Since our approach is to analyze fluid systems using the approximate velocity from the optical flow algorithms, we have to compute the time-dependent velocity fields using a sequence of images or a movie of the observed system. Some recent work has been done in fluid flow estimation [91–94] using optical flow techniques developing new approaches with promising results. In most of these new approaches, the authors concentrate on the modification of the data fidelity, the regularization term and the numerical scheme of the optical flow algorithm. When we compute the fluid flows, the optical flow algorithms are applied to each consecutive pair of images in the observed image sequence of the system. However, these algorithms do not impose any characteristic between two time-adjacent flow fields which we considered to develop a new algorithm to compute fluid flows using an image sequence.

To develop the new algorithm, we use the stream function formulation that we

developed in [36, 37] and presented in Chapter 3. Since both the stream function formulation and the potential function formulation have a similar development, we chose the stream function formulation to construct the new algorithm. In the development of the algorithm, we focused on the lack of temporal relationships between two time-adjacent flow fields and we include a new term to the energy functional by defining a new characteristic between two time-adjacent vector fields via the stream function [95]. In other words, defining the new characteristic can be considered as the regularization of the energy functional in the time direction. Since we compute the vector fields for multiple time steps imposing the time regularity, the new method is named as the Multi-Time Step Method.

6.1 Multi-Time Step Method

Here we introduce the multi-time step method of computing optical flow for a sequence of time-dependent images through computing n time-dependent vector fields simultaneously. Before we develop the multi-time step method, we need to select an optical flow energy functional that performs well on fluid flows. According to the qualitative analysis in Chapter 3 and the quantitative analysis in Chapter 4, the stream function formulation with conservation of intensity data fidelity and smoothness regularization term outperforms the other methods in incompressible fluids and the potential function formulation with the same data fidelity and the regularization provides comparable results on the compressible fluids. Therefore, we explain the construction of the multi-time step method on incompressible fluids using the stream function formulation with the energy functional

$$E(\psi) = \int_{\Omega} \left(I_t - I_x\psi_y + I_y\psi_x\right)^2 \, d\Omega + \alpha \int_{\Omega} \left(\psi_{xx}^2 + \psi_{yy}^2 + \psi_{xy}^2 + \psi_{yx}^2\right) d\Omega. \tag{6.1}$$

Recall that the Euler-Lagrange equation for the functional in Eq.(6.1) is obtained in Chapter 3 as

$$[2A^*(I_t + A) + \alpha(B + B^*)]\,\psi = -2A^* I_t, \tag{6.2}$$

where the operators A and B can be written as

$$A = -I_x D_y + I_y D_x \text{ and } B = D_{xx} D^*_{xx} + D_{yy} D^*_{yy} + D_{xy} D^*_{xy} + D_{yx} D^*_{yx}$$

respectively and the derivative operators are defined as in Chapter 3. Note that, in Chapter 3 we used A and B with different subscripts for the operators of different data fidelities and the regularization terms, but here we denote the operators of data fidelity and the regularization term by A and B for symbolic convenience. When we compute one velocity field at a time, there is no difference between the multi-time step method and the usual stream function method. Therefore, the energy functional for the multi-time step method when the step size is represented as $n = 1$ in Eq.(6.1). The solution for the functional in Eq.(6.1) is obtained by solving the Euler-Lagrange equation in Eq.(6.2) for the stream function ψ. As explained in Chapter 3 we use LU-factorization and Gaussian elimination to solve the system.

When we consider $n = 2$ or more, we include an additional term in the data fidelity by introducing regularity in the time direction by assuming that two consecutive stream functions have similar behavior. Suppose we are given T time-adjacent images as a movie of a dynamical system, then evolving the window slow enough that considerations of continuously evolving frame views allow inference of the underlying dynamical systems even though the flow is unsteady. In other words assume $I(x, y, t)$ is continuous with respect to t throughout the scene. In fact we cope with this assumption by including a new term with a weighting factor. For instance, if there are

only two stream functions ψ_1 and ψ_2, the additional minimizing integral would be

$$\int_\Omega (\psi_1 - \psi_2)^2 \, d\Omega \tag{6.3}$$

added to the chosen energy functional already designed for assumed prior information. In this case we use three images at a time to compute the flow in two different time instances and here ψ_1 is the stream function governing between image 1 and image 2 and ψ_2 is the stream function between image 2 and image 3. To obtain the complete energy functional for the multi-time step method with $n = 2$, we first obtain two individual energy functionals for ψ_1 and ψ_2 as

$$\begin{aligned} E_1(\psi_1) &= \int_\Omega (I_{1t} - I_{1x}\psi_{1y} + I_{1y}\psi_{1x})^2 \, d\Omega + \alpha \int_\Omega \left(\psi_{1xx}^2 + \psi_{1yy}^2 + \psi_{1xy}^2 + \psi_{1yx}^2\right) d\Omega \\ E_2(\psi_2) &= \int_\Omega (I_{2t} - I_{2x}\psi_{2y} + I_{2y}\psi_{2x})^2 \, d\Omega + \alpha \int_\Omega \left(\psi_{2xx}^2 + \psi_{2yy}^2 + \psi_{2xy}^2 + \psi_{2yx}^2\right) d\Omega \end{aligned}$$

and then we combine the new term in Eq.(6.3) with $E_1(\psi_1)$ and $E_2(\psi_2)$ by a weighting factor $\beta > 0$. Here I_1 and I_2 are the intensities corresponding to the first pair of images and the pair of image 2 and image 3 respectively. The resulting energy functional is given by

$$\begin{aligned} E(\psi_1, \psi_2) = \int_\Omega \Big(I_{1t} - I_{1x}\psi_{1y} + I_{1y}\psi_{1x})^2 + (I_{2t} - I_{2x}\psi_{2y} + I_{2y}\psi_{2x}\Big)^2 \, d\Omega \\ + \beta \int_\Omega (\psi_1 - \psi_2)^2 \, d\Omega \\ + \alpha \int_\Omega (\psi_{1xx}^2 + \psi_{1yy}^2 + \psi_{1xy}^2 + \psi_{1yx}^2) d\Omega \\ + \alpha \int_\Omega (\psi_{2xx}^2 + \psi_{2yy}^2 + \psi_{2xy}^2 + \psi_{2yx}^2) d\Omega. \end{aligned} \tag{6.4}$$

Thus we have the energy functional for two time instances at once to emphasize the time regularity as well as the spatial regularity.

Now taking the Gateaux derivative as in Eq.(2.20) of the functional in Eq.(6.4), the Euler-Lagrange equations corresponding to ψ_1 and ψ_2 are then the system of

PDEs as

$$\begin{aligned}[2A_1^*A_1 + 2\beta(\psi_1 - \psi_2) + \alpha(B + B^*)]\psi_1 &= -2A_1^*I_{1t} \\ [2A_2^*A_2 - 2\beta(\psi_1 - \psi_2) + \alpha(B + B^*)]\psi_2 &= -2A_2^*I_{2t},\end{aligned} \tag{6.5}$$

where

$$A_1 = (-I_{1x}D_y + I_{1y}D_x) \text{ and } A_2 = (-I_{2x}D_y + I_{2y}D_x)\,.$$

We can generalize the energy functional for n stream functions at n successive time instances as shown in the following:

$$\begin{aligned}E(\psi_1, \psi_2, ..., \psi_n) = \sum_{k=1}^{n} \int_\Omega (I_{kt} - I_{kx}\psi_{ky} + I_{ky}\psi_{kx})^2 \, d\Omega \\ +\beta \sum_{k=1}^{n-1} \int_\Omega (\psi_k - \psi_{k+1})^2 \, d\Omega \\ +\alpha \sum_{k=1}^{n} \int_\Omega (\psi_{kxx}^2 + \psi_{kyy}^2 + \psi_{kxy}^2 + \psi_{kyx}^2) d\Omega.\end{aligned} \tag{6.6}$$

Similar to $n = 2$, the Euler-Lagrange equations can be obtained for any number n, using the Gateaux derivative as in Eq. (2.20) of the functional in Eq. (6.6) with respect to $\psi_1, \psi_2, ...,$ and ψ_n. Further, we set the regularization parameter α as 2α for convenient simplification. The Euler-Lagrange equations for any integer n which is a system of n partial differential equations is obtained as

$$A_1^*(I_{1t} + A_1\psi_1) + \beta(\psi_1 - \psi_2) + \alpha(B + B^*)\psi_1 = 0$$

$$A_k^*(I_{kt} + A_k\psi_k) + \beta(-\psi_{k-1} + 2\psi_k - \psi_{k+1}) + \alpha(B + B^*)\psi_k = 0, \quad \text{for } k = 2, 3, ...n - 1$$

$$A_n^*(I_{nt} + A_n\psi_n) - \beta(\psi_{n-1} - \psi_n) + \alpha(B + B^*)\psi_n = 0,$$

where

$$A_k = (-I_{kx}D_y + I_{ky}D_x) \quad \forall k.$$

The above set of Euler-Lagrange equations can be reformulated as a linear system as

$$[K + \alpha L]\, z = b \tag{6.7}$$

where

$$K = \begin{bmatrix} A_1^* A_1 + \beta & -\beta & & & & \\ -\beta & A_2^* A_2 + \beta & -\beta & & & \\ \ddots & \ddots & \ddots & & & \\ & & -\beta & A_{n-1}^* A_{n-1} + \beta & -\beta & \\ & & & -\beta & A_n^* A_n + \beta & \end{bmatrix},$$

$$L = \begin{bmatrix} B + B^* & & & & \\ & B + B^* & & & \\ & \ddots & & & \\ & & & B + B^* & \\ & & & & B + B^* \end{bmatrix}, \quad z = \begin{bmatrix} \psi_1 \\ \psi_2 \\ \vdots \\ \psi_{n-1} \\ \psi_n \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} A_1^* I_{1t} \\ A_2^* I_{2t} \\ \vdots \\ A_{n-1}^* I_{(n-1)t} \\ A_n^* I_{nt} \end{bmatrix}.$$

All the entries of the matrices K and L are block matrices of the size $m \times m$, where $m = pq$ and p and q are the dimensions of the image domain. The entries of the vectors z and b are vectors of the size $m \times 1$.

To solve the above system, first we take the LU decomposition of the matrix $K + \alpha L$ and then use Gaussian elimination. The solution of the above system yields $\psi_1(x, y) \equiv \psi_1(x, y, t_1)$, $\psi_2(x, y) \equiv \psi_2(x, y, t_2)$, ..., $\psi_n(x, y) \equiv \psi_n(x, y, t_n)$. Taking the

Hamiltonian gradient on each of them separately, the vector fields for each time instance, $t_1, t_2, ..., t_n$ can be obtained. That is, the velocity components $u_1, u_2, ..., u_n$ and $v_1, v_2, ..., v_n$ are computed from $\psi_1, \psi_2, ..., \psi_n$ using $\langle u_k, v_k \rangle = \langle -\psi_{ky}, \psi_{kx} \rangle$ for $k = 1, 2, ...n$.

Another advantage of the multi-time step method is that we can use higher-order finite difference approximations to compute the time derivative of image intensity, I_t instead of the forward difference approximation necessary when just two images are available. This can be done as we have a sequence of images rather than two images. For instance, if we use third-order finite difference approximation to compute I_t, then five time adjacent images are considered to compute I_t. This will help to further improve the continuity of the flow in the time direction. In addition to the above advantages, another positive consequence is that the multi-time step method computes n vector fields at a time, but the negative consequence of this is computationally expensive.

The multi-time step method can be expressed in terms of any data fidelity in Chapter 3 and any regularization term in Chapter 4. For instance, if we use the continuity equation data fidelity Eq.(3.11) and the second-order div-curl regularization, we have to substitute

$$A = -I_x D_y + I_y D_x \text{ and } B = D_{xx} D^*_{xx} + D_{yy} D^*_{yy} + D_{xy} D^*_{xy} + D_{yx} D^*_{yx}$$

in the multi-time step method algorithm.

For compressible fluids, we have to apply the potential function formulation and energy functional corresponding to the conservation of image intensity data fidelity and the smoothness regularization term is obtained as

$$E(\phi) = \int_\Omega \left(I_t + I_x \phi_x + I_y \phi_y\right)^2 \, d\Omega + \alpha \int_\Omega \left(\phi^2_{xx} + \phi^2_{yy} + \phi^2_{xy} + \phi^2_{yx}\right) d\Omega. \tag{6.8}$$

The multi-time step method in terms of the potential function formulation for n potential functions can be obtained as

$$E(\phi_1, \phi_2, ..., \phi_n) = \sum_{k=1}^{n} \int_\Omega \left(I_{kt} + I_{kx}\phi_{kx} + I_{ky}\psi_{ky}\right)^2 \, d\Omega$$
$$+\beta \sum_{k=1}^{n-1} \int_\Omega \left(\phi_k - \phi_{k+1}\right)^2 \, d\Omega$$
$$+\alpha \sum_{k=1}^{n} \int_\Omega (\phi_{kxx}^2 + \phi_{kyy}^2 + \phi_{kxy}^2 + \phi_{kyx}^2) d\Omega. \tag{6.9}$$

The Euler-Lagrange equations for the functional in Eq.(6.9) with respect to $\phi_1, \phi_2, ...,$ and ϕ_n. are obtained as

$$A_1^*(I_{1t} + A_1\phi_1) + \beta(\phi_1 - \phi_2) + \alpha(B + B^*)\phi_1 = 0$$

$$A_k^*(I_{kt} + A_k\phi_k) + \beta(-\phi_{k-1} + 2\phi_k - \phi_{k+1}) + \alpha(B + B^*)\psi_k = 0, \quad \text{for } k = 2, 3, ...n-1$$

$$A_n^*(I_{nt} + A_n\phi_n) - \beta(\phi_{n-1} - \phi_n) + \alpha(B + B^*)\phi_n = 0,$$

where

$$A_k = (I_{kx}D_x + I_{ky}D_y) \quad \forall k.$$

Solving these Euler-Lagrange equations for $\phi_1, \phi_2, ...,$ and ϕ_n is similar to the stream function method. The vector fields are obtained by taking the gradients of the computed potential functions as $\langle u_k, v_k \rangle = \langle \phi_{kx}, \phi_{ky} \rangle$ for $k = 1, 2, ...n$. Similar to the stream function formulations, the potential function method is also extended for any data fidelity or any regularization term.

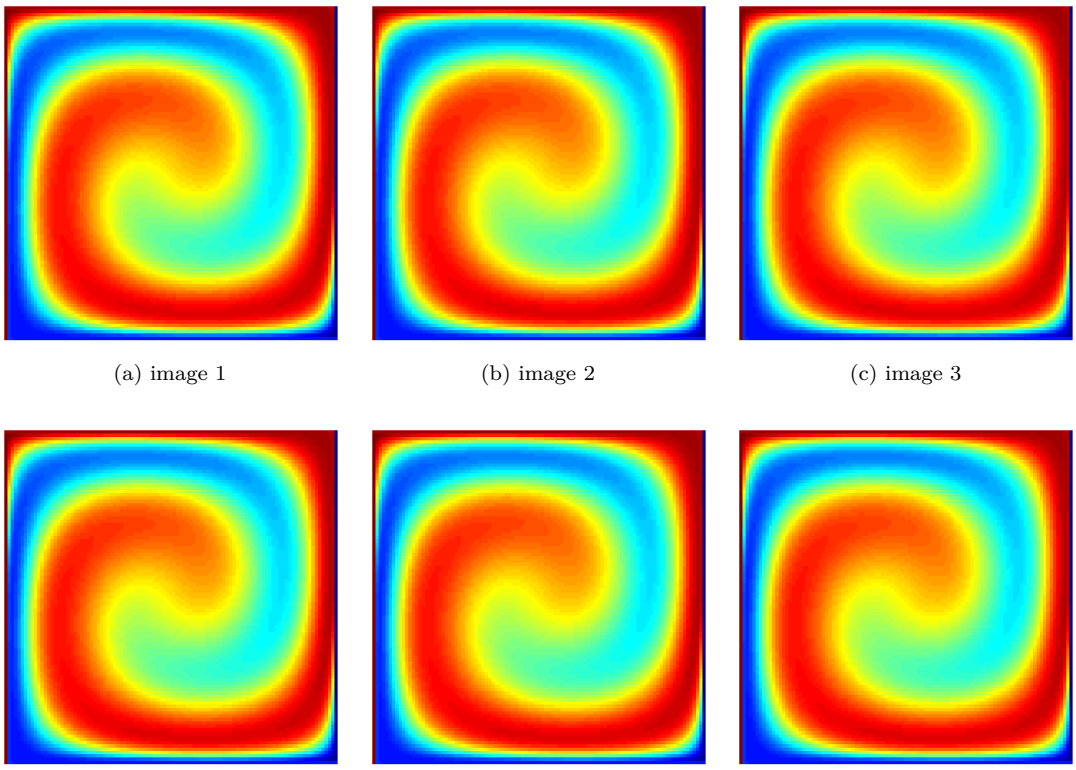
6.2 Results from Multi-Time Step Method

In this section, we will demonstrate the performance of the multi-time step methods which are developed using the energy functionals in Eq.(6.1) and Eq.(6.8). Also we present the improvement of the accuracy of our algorithm with larger n. For the above purposes, we use two benchmark data sets, the gyre and the source data sets. In addition to these two data sets, we use an oceanic data set introduced in Chapter 3. When we compare the reconstructed vector fields with the true vector field, we need a figure of merit for comparison. Therefore, we compute angular error between the computed flow and the true flow and then the mean over the domain to obtain mean angular error [68] as explained in Chapter 4.

Fig. 6.1 shows 6 consecutive images produced from the vector field in Eq.(2.34) by evolving according to the continuity equation in Eq.(2.29). The true flow field and two other images are shown in images (c), (a) and (b) respectively, in Fig. 2.4. When we apply the multi-time step method with step size $n = 4$, with fourth-order finite difference approximation for I_t, we need eight images to compute the vector fields. We have shown the first six images of the considered sequence in the Fig. 6.1.

Note that the synthetic images we use in this chapter are different from the images shown in Chapter (2), even though the true velocity field is the same. Due to these changes in the images, the MAE values may differ from the MAE values in Chapter 3.

Since the data set is divergence free, the multi-time step method in stream function formulation is applied to compute the velocity fields on the gyre data set. Our first step is to compare the results by varying the step size n, the number of stream functions ψ computed at a given time using multiple images. We applied our multi-time step method on an image sequence of the gyre data set as shown in Fig. 6.1, for different n values. For instance, if we apply a first-order finite difference approximation to



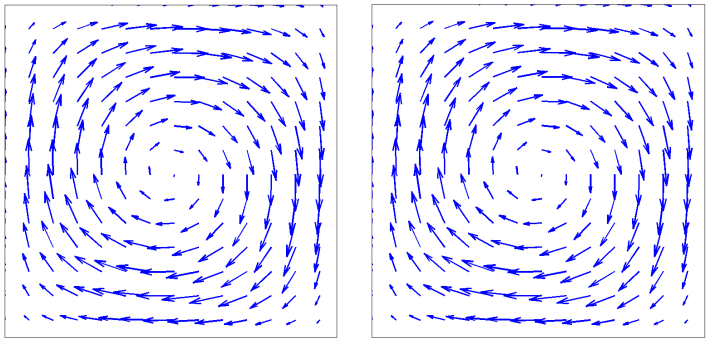
(a) image 1 (b) image 2 (c) image 3

(d) image 4 (e) image 5 (f) image 6

Figure 6.1: Gyre image sequence – Images (a) - (f) show 6 time-adjacent images of the gyre image sequence. These images are generated as we explained in Chapter (2). Two other images and the true vector field are shown in Fig. 2.4

estimate I_t with the step size $n = 1$, then we compute one stream function ψ_1 using two images, image 1 and image 2. The velocity components $(u_1, v_1) = (-\psi_{1y}, \psi_{1x})$ represent the motion field between image 1 and image 2. If however we choose the step size $n = 2$, then we compute two stream functions ψ_1 and ψ_2 using three images. Then $(u_1, v_1) = (-\psi_{1y}, \psi_{1x})$ is the motion field between image 1 and image 2 while $(u_2, v_2) = (-\psi_{2y}, \psi_{2x})$ is the motion field between image 2 and image 3 respectively. Continuing in this manner, we can increase the step size n. Note that if there is a sequence of nine images, we can compute eight vector fields for each consecutive image pair. When the step size $n = 1$, eight separate computations are necessary but when $n = 2$, only four computations are necessary and so on. Again we emphasize that the advantage of choosing a larger n is that the time regularity is emphasized

as seen clearly in the computations. Including more terms of the form in Eq.(6.3) penalizes large changes of ψ between successive frames.



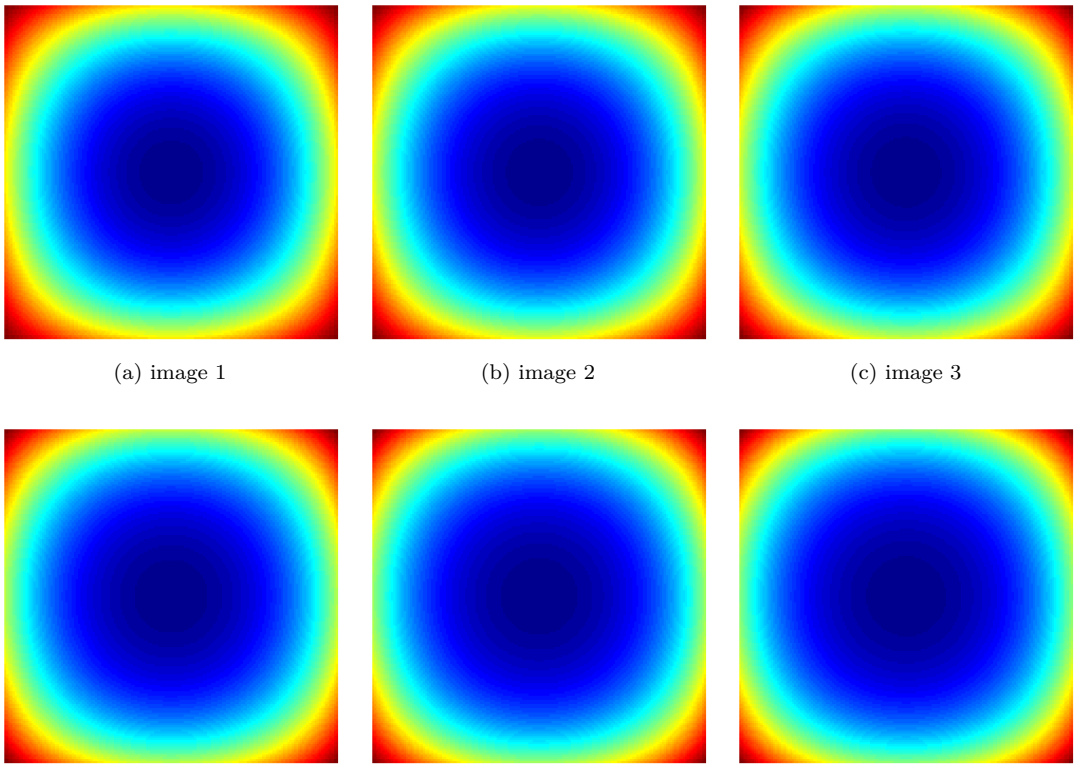
(a) $n = 1$ (b) $n = 2$

(c) $n = 3$ (d) $n = 4$

Figure 6.2: Gyre flow from Multi-Time Step Method – Images (a), (b), (c), and (d) show vector fields computed on the image (c) in Fig. 6.1 by the multi-time step method with $n = 1$, 2, 3, and 4 respectively. While all estimated vector fields are visually similar, the mean angular error improve up to $n = 3$.

After applying the multi-time step method on the gyre data set, the results for $n = 1, 2, 3$ and 4 on the image (c) in Fig. 6.1 are shown in images (a), (b), (c), and (d) in Fig. 6.2, respectively. The mean angular errors for $n = 1, 2, 3$, and 4 are 0.9837°, 0.9826°, 0.9807° and 0.9883° respectively. In this example, the accuracy of the algorithm improves until $n = 3$. In all the above reconstructions, the regularization parameter α was selected so that it minimizes the MAE and the parameter β was fixed at $\beta = 0.01$.

Our next example is the source data set where we demonstrate that the multi-time step method performs well even with the larger step size with the potential function formulation. A sequence of six images is shown in the Fig. 6.3 which we constructed evolving the velocity field in Eq.(2.36) according to the continuity equation in Eq.(2.29) as we explained in Chapter 2.



(a) image 1 (b) image 2 (c) image 3

(d) image 4 (e) image 5 (f) image 6

Figure 6.3: Source image sequence – Images (a) - (f) show 6 time-adjacent images of the source data image sequence. These images are generated as we explained in Chapter 2. Two other images and the true vector field are shown in Fig. 2.5.

Since the source flow is not divergence free, the appropriate algorithm is the multi-time step method in potential function formulation for the optical flow computation. We applied the algorithm on a sequence of source images and computed the flow on images 1 - 6 as shown in Fig. 6.3. We compute the mean angular error of the computed flow corresponding to each of the images with the step size $n = 1, 2, 3$ and 4 as shown in Table 6.1. Each column represents the mean angular errors from the step sizes

$n = 1, 2, 3$ and 4 for the six images. It can be observed that the mean angular errors are improving with the step size until $n = 3$ and then breaks. However, the mean angular error of the computed flow on image 4 improves even with the step size $n = 4$.

Table 6.1: The MAE values for the six source images shown in Fig. 6.3 are computed. In the computation, the multi-time step method in potential function formulation was applied and flow fields were computed for step sizes $n = 1, 2, 3$ and 4. The first row represents the MAE values for six images for $n = 1$ and the second row is for $n = 2$ etc. In general, MAE improves until $n = 3$ and then becomes unpredictable. However, the MAE corresponding to the flows on image 4 improves even after step size $n = 3$.

Step size	Images in which the flow is computed					
	image1	image 2	image 3	image 4	image 5	image 6
$n = 1$	2.664°	2.693°	2.685°	2.752°	2.671°	2.723°
$n = 2$	2.601°	2.642°	2.623°	2.691°	2.630°	2.692°
$n = 3$	2.572°	2.565°	2.570°	2.592°	2.567°	2.594°
$n = 4$	2.590°	2.565°	2.589°	2.590°	2.595°	2.604°

The graphical representation of the mean angular errors verses step sizes for six different images are shown in Fig. 6.4. It is clearly visible that the mean angular errors improve up to $n = 3$ and then become unstable. On the other hand, even though the images are different and the MAEs are not the same for each step size, when $n = 3$ and $n = 4$, we can see that the MAEs are close to each other. The mean angular errors corresponding to $n = 3$ and $n = 4$ are shown in red and magenta colors respectively in Fig. 6.4 and the two curves are almost flat. This may be due to the regularity in the time direction.

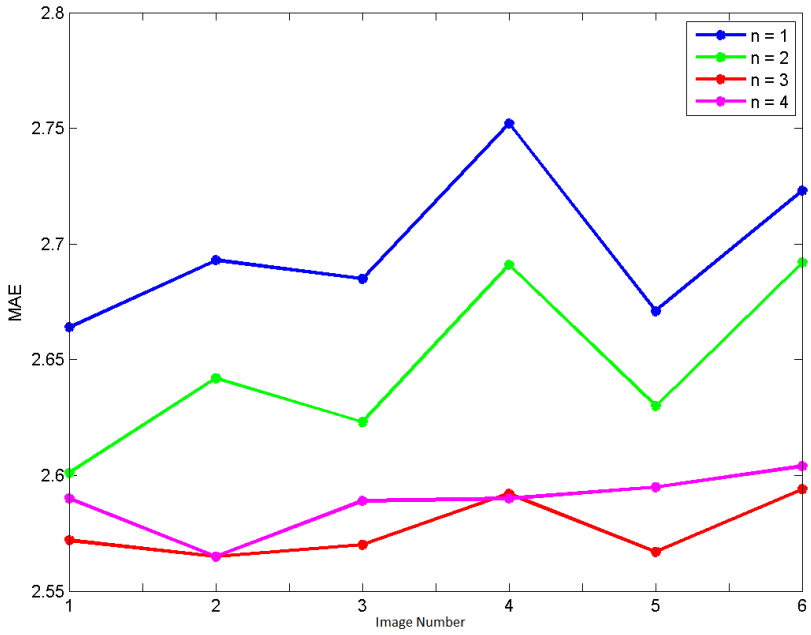
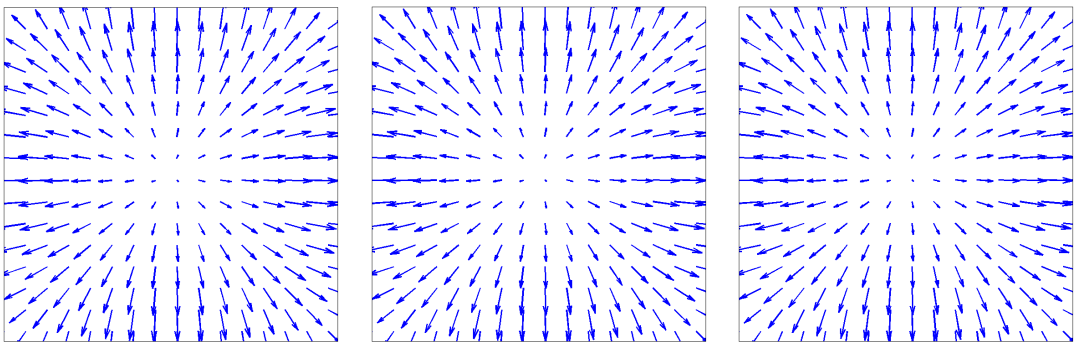
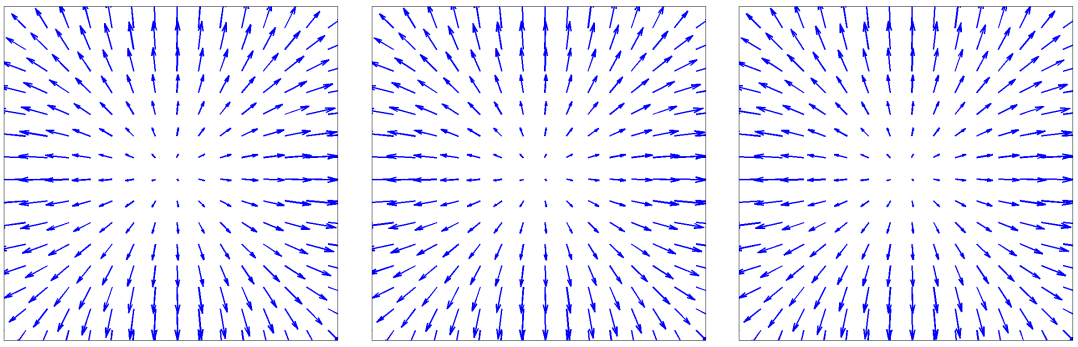


Figure 6.4: MAE on six images for different n – The graph shows the mean angular error for the computed flow by changing the step size n on the six images shown in Fig. 6.3. The blue and green curves represent the step size $n = 1$ and $n = 2$ and the mean angular errors are relatively high. The red and magenta color curves are for $n = 3$ and $n = 4$ and their mean angular errors are lower as well as close to each other along the image sequence because of the more regularity in time direction.

According to the results, the step size $n = 3$ produces relatively better solutions on the six images shown in Fig. 6.3. We include the corresponding flow fields for each image in Fig. 6.5. It is hard to find any qualitative differences on these images, but they have quantitative differences as shown in Table 6.1. The best MAE is obtained from the flow on the image (b) and it is equal to 2.565°.



(a) flow on image 1 (b) flow on image 2 (c) flow on image 3

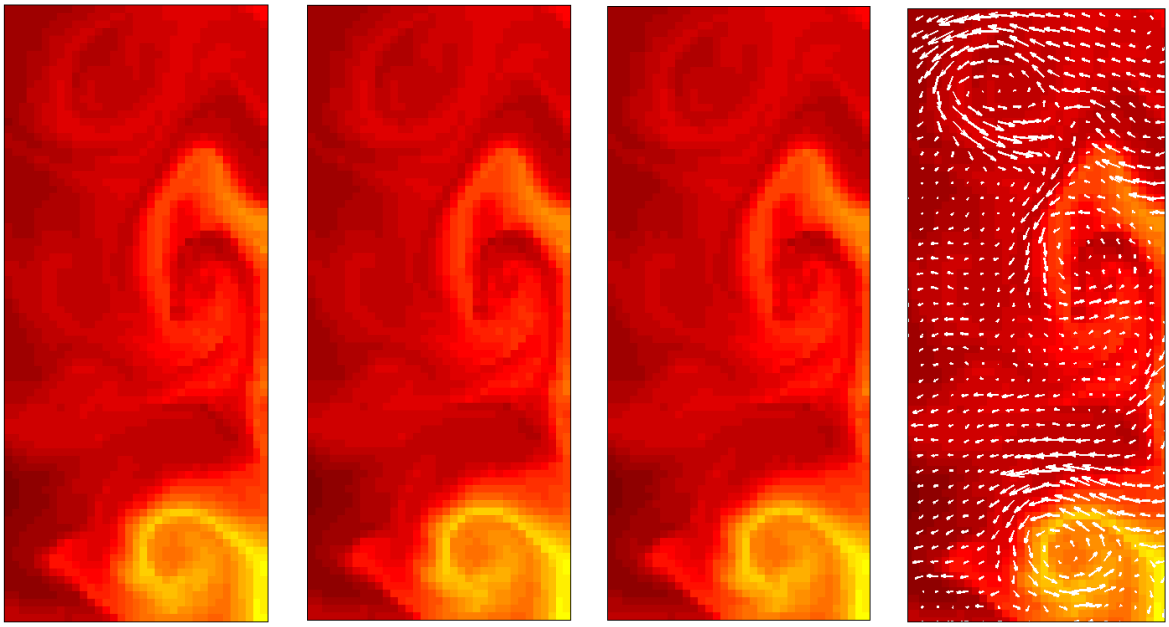


(d) flow on image 4 (e) flow on image 5 (f) flow on image 6

Figure 6.5: Source flow fields with $n = 3$ – Images (a) - (f) show 6 time-adjacent velocity fields computed on the six images in Fig. 6.3. In the computation, we used multi-time step method in potential function formulation with $n = 3$. Flow field in image (b) produces the minimum mean angular error of 2.565°. However, qualitative differences of the flow fields are hardly visible.

6.2.1 An Oceanographic Data Set

Now we apply the algorithm to a natural scenario which shows the sea surface temperature off the coast of Oregon, USA. This data set was generated from a 3-D ocean model, using data obtained from Geostationary Operational Environmental Satellite (GOES) referred to in [96] as we explained in Chapter 3. In Fig. 6.6, images (a), (b) and (c) show sea surface temperature data of three consecutive hours on August 1, 2002. The image (d) represents the true vector field of the mixing temperature corresponding to image (b).

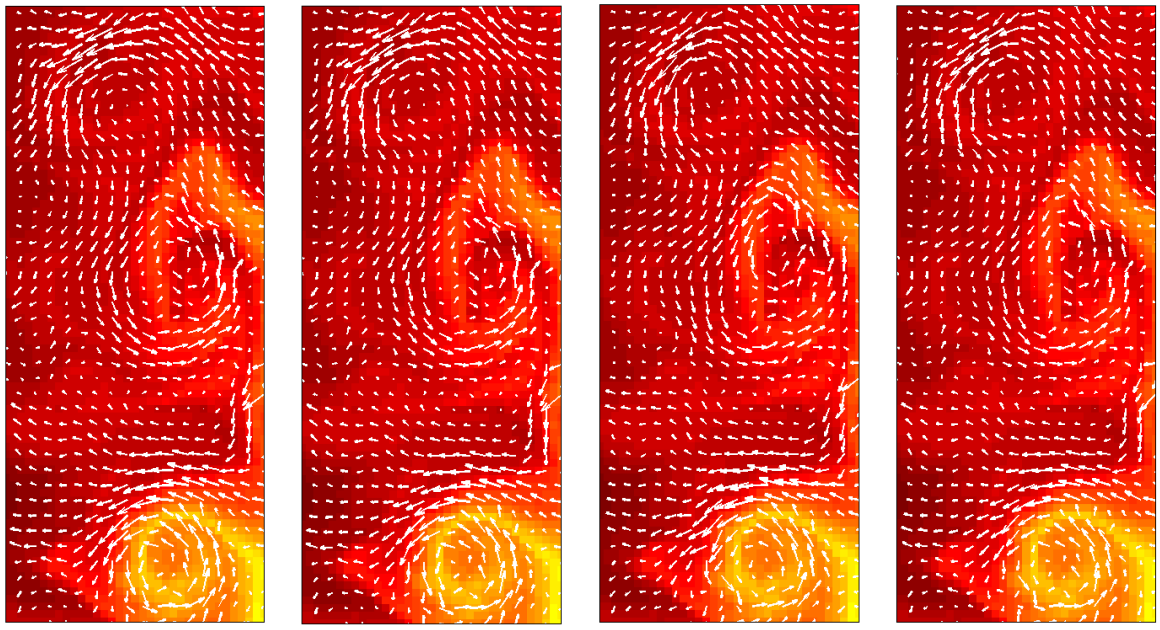


(a) (b) (c) (d)

Figure 6.6: SST data and true flow – Three consecutive images of the SST data set are shown in (a), (b) and (c) respectively. The flow on image (b) is shown in the image (d)

Since we have a time-dependent sequence of images of the SST data, we can apply the multi-time step method to compute the vector fields. When the step size is n, we compute n consecutive vector fields at a time and we require $n + 4$ images, if we apply fourth-order finite difference approximations to compute I_t. The Fig. 6.7 represents the computed flow fields on the image (b) shown in Fig. 6.6 with the step size $n = 1, 2, 3$ and 4 in images (a), (b), (c), and (d) respectively. In each case, the algorithm captures the gyres accurately and it is clearly visible that when $n = 3$, the algorithm captures the laminar flow as seen just above the bottom gyre. Except for the above laminar flow, all the other vectors represent similar behavior and the differences between flow fields comes from different step sizes are not visible. Now we can compare the two vector fields consisting of large numbers of vectors by comparing single numbers. In the comparison of the step sizes; we use the percentage mean

angular error.



(a) (b) (c) (d)

Figure 6.7: SST Flow – The computed flow fields for the data showed in 6.6 with n equals to 1, 2 and 3 are shown in (a), (b), (c) and (d) respectively. While all these are roughly similar and so not immediately different to visual inspection, there are visible differences appear upon closer inspection.

Fig. 6.8 shows the graph of percentage mean angular error of the computed flow relative to the true flow versus the step size used to compute the flow fields. According to the graph, the percentage mean angular error fluctuates and the minimum is achieved when $n = 3$ as we can see in the flow fields. Note that for all the reconstructions in Fig. 6.6, the regularization parameter α was selected so that it minimizes the mean angular error and the parameter β was fixed at $\beta = 0.01$.

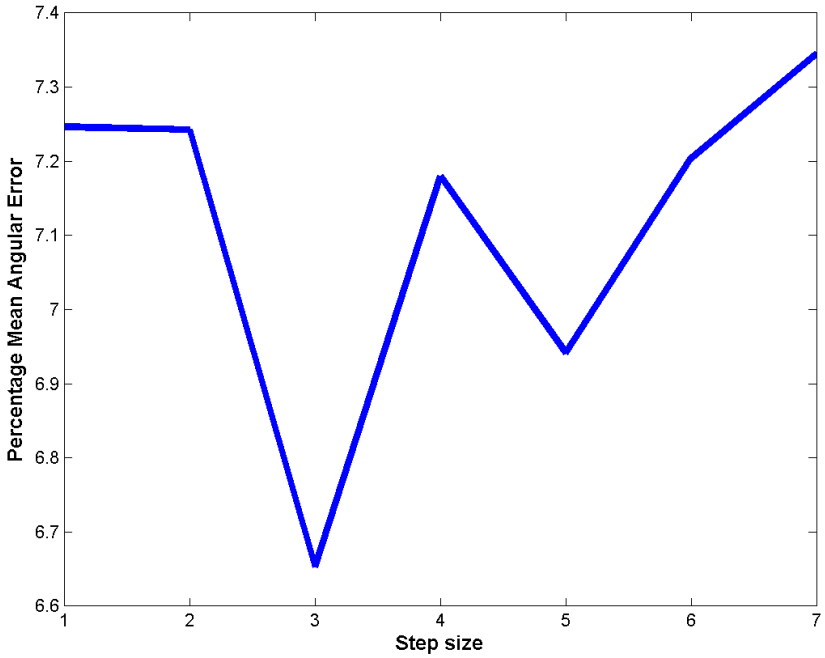
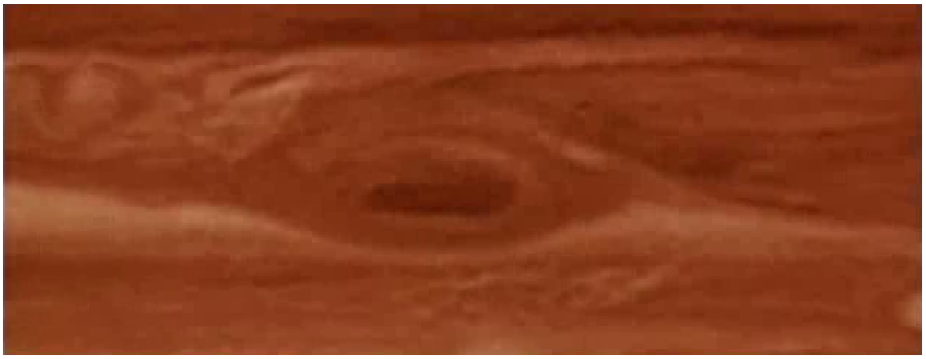


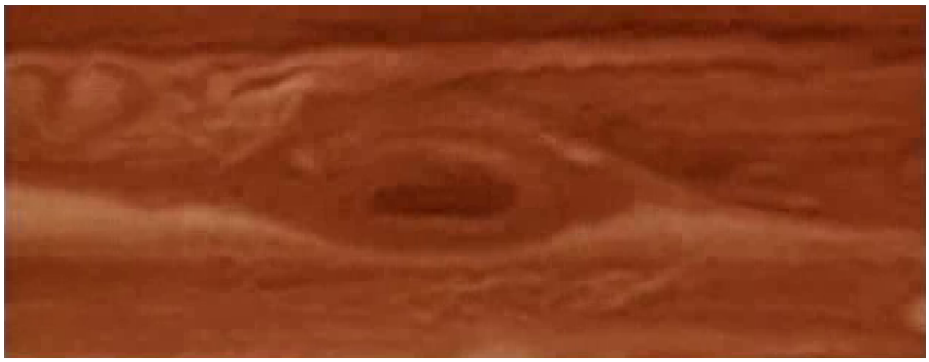
Figure 6.8: Percentage MAE vs Step size – The graph shows the percentage of the mean angular error for the computed flow by changing the step size n on the image (b) shown in Fig. 6.6. For the specified parameters at $n = 3$, multi-time step method is best overall.

6.2.2 A Planetary Data Set

Our final application of the multi-time step method is on Jupiter's atmospheric data. The data set was taken by the spacecraft Voyager 2 with the time step of one Jovian day. That is, the images were taken every 9.94 hours.



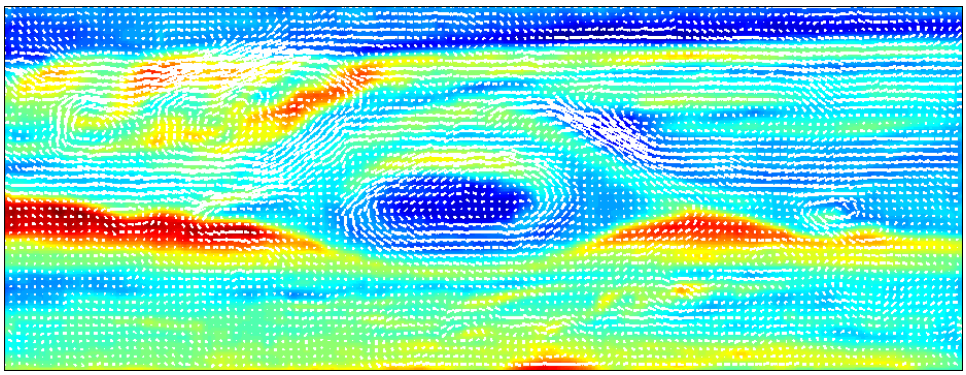
(a)



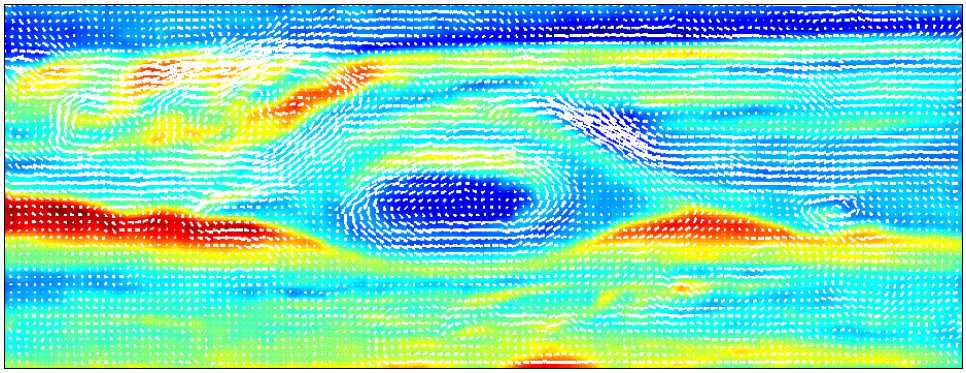
(b)

Figure 6.9: GRS images [97] – Two consecutive images captured by the spacecraft Voyager 2 are shown in images (a) and (b) respectively. The two images are one Jovian day apart.

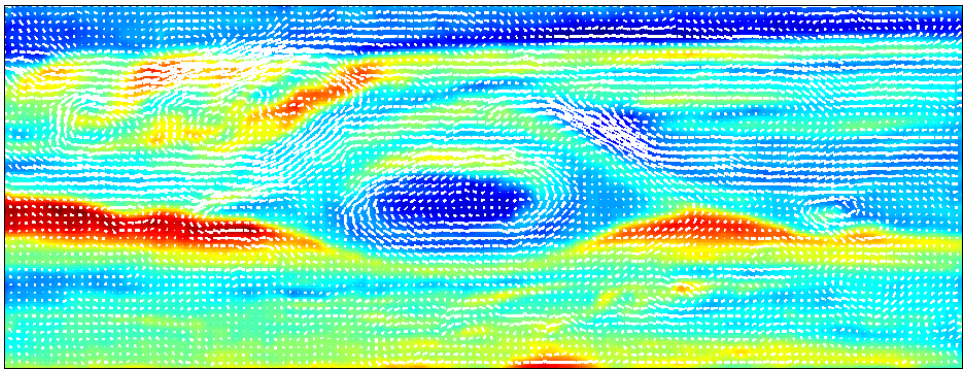
We apply the multi-time step method on the images which we displayed in Fig. 6.9 for $n = 1, 2$ and 3. In Fig. 6.10, the computed flow fields from multi-time step with $n = 1, 2$ and 3 are shown in images (a), (b) and (c) respectively. In all three cases, the algorithms capture the boundaries of the vortex reasonably, especially the top of the vortex. When $n = 3$, the algorithm is able to capture the bottom of the vortex.



(a) flow with $n = 1$



(b) flow with $n = 2$



(c) flow with $n = 3$

Figure 6.10: Multi-time step flow for the GRS – images (a) - (c) show the computed flow from the multi-time step method on the image (a) in Fig. 6.9 with step sizes $n = 1, 2$ and 3, respectively. Resulting flow fields from each step size are reasonable, especially around the GRS. When the step size increases, the flow on the lower part of the GRS captures the structure.

Chapter 7

Quasi-Static Equations with Coriolis Force in Optical Flow Method

As we know, the planets in our solar system rotate about their respective axis while revolving about the sun. Due to the rotation of planets, the coriolis force may affect to the motion of fluid systems on the planets, if the system is observed from a rotating frame of reference. Most of the satellites observing planets also revolve and rotate according to the corresponding planet. When a system is observed from a rotating satellite and the observed images are used to compute velocity fields, we must take into account the coriolis effect. We make this correction before completing the computation of vector fields by modifying the energy functional.

For instance, in the atmosphere, air flows in a straight line from areas of high pressure to areas of low pressure. However, due to the coriolis force, this straight path becomes curved. When we deal with systems having geostrophic[1] flows of this type, the system can be represented by the quasi-static equations [47] which are valid in the

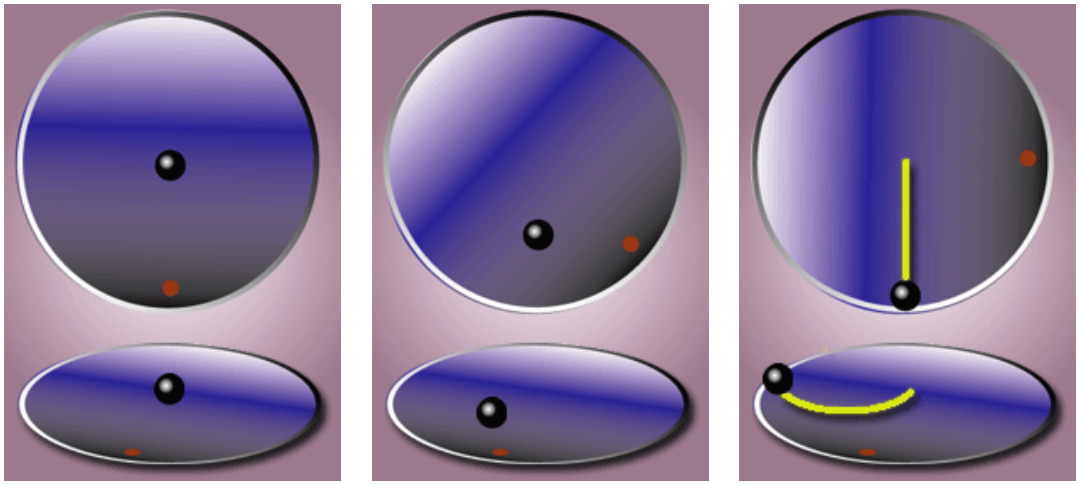
[1]A flow is called geostrophic, if the the pressure gradient force and the Coriolis force are balance with negligible friction force

absence of friction and diffusion. However, diffusion is not easy to ignore as we can not expect divergence free fluid motion. For this reason, when we develop the energy functional using quasi-static equations we add a diffusion term as the regularization term. The quasi-static equations are represented in Eq.(7.2). However, their present formulation is not ready to directly include in an optical flow energy functional to be solved for determining velocity fields. Some terms such as pressure, density and vertical velocity components in quasi-static equations are unable to incorporate in optical flow energy functional as the images do not provide such information. Therefore, we narrow the equations by eliminating the terms which are not related to optical flow computation. In this way, we are able to solve the problem in a convenient way while we expect the simplified problem leads to a useful solution estimating that leads to solution of the actual problem. First we give an introduction of the coriolis force in Sec. 7.1 and then explain the quasi-static equations and their adaptation of an optical flow algorithm in the rest of this chapter.

7.1 Coriolis Force

Objects such as wind storms and ocean currents are deflected from a straight path relative to a rotating reference frame by the coriolis effect due to the so called coriolis force. As an example on a global scale, the surface of the Earth is a rotating reference frame and in the northern hemisphere the deflection is to the right while in the southern hemisphere the deflection is to the left. However, there is no deflection at the equator.

Fig. 7.1 shows three different time instances of motion of an object in the upper disk of the each image. The object is the black dot and an observer is the red dot. Even though the object is moving in a straight path with respect to an inertial reference frame, the observer thinks that the object moves on a curved path. This phenomenon



(a) (b) (c)

Figure 7.1: Coriolis effect – The upper disks of all three images represent the inertial frame of reference and the lower disks represent the rotating frame of reference. The black ball is an object and the red dot is the observer. Even though the object moves in a straight line with respect to the inertial frame of reference, the observer thinks the object moves in a curved path. The initial position of object is shown in image (a), a midway position is shown in image (b) and the final position is shown in image (c). A complete explanation with a movie is available in [98]

occurs due to the coriolis force. On a rotating planet, for a given latitude φ with the angular velocity ω, the coriolis parameter is

$$f = 2\omega \sin \varphi. \tag{7.1}$$

This implies that the value of the coriolis parameter is positive in the northern hemisphere and negative in the southern hemisphere. The coriolis parameter is zero at the equator as the latitude φ is zero. The Coriolis force is always perpendicular to the flow and hence the direction is $\frac{\pi}{2}$ radians to the right in the northern hemisphere and left in the southern hemisphere. In Sec. 7.2, we are going to adopt the quasi-static equations to suit the optical flow environment so that we can incorporate the coriolis force in an optical flow algorithm.

7.2 Quasi-Static Equations

In the absence of friction and diffusion, the quasi-static equations as in [47] are given in the form

$$\frac{du}{dt} - fv = -\frac{1}{\rho_0}\frac{\partial p'}{\partial x} \tag{7.2a}$$

$$\frac{dv}{dt} + fu = -\frac{1}{\rho_0}\frac{\partial p'}{\partial y} \tag{7.2b}$$

$$0 = -\frac{\partial p'}{\partial z} - \rho' g \tag{7.2c}$$

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0 \tag{7.2d}$$

$$\frac{\partial \rho'}{\partial t} + u\frac{\partial \rho'}{\partial x} + v\frac{\partial \rho'}{\partial y} + w\frac{\partial \bar{p}}{\partial z} = 0, \tag{7.2e}$$

where u, v and w are the velocity components along the x, y and z directions, p is pressure and ρ is density. These set of equations are usually called the "primitive equations" in the field of atmospheric dynamics and they are basically the Euler-equations for gas dynamics, expressed in Cartesian geometry on a rotating reference frame. Eq.(7.2a) and Eq.(7.2b) show the momentum in the x and y directions, where traditionally x represents east and y represents north. These two equations are derived by applying Newton's second law, which states that the sum of all forces acting on a unit mass is equal to the acceleration of the unit mass. Eq.(7.2c) is the vertical momentum (hydrostatic approximation) which represents the balance between pressure gradient and gravity. Eq.(7.2d) represents the continuity of the fluid mass. The last equation, Eq.(7.2e), represents the Thermodynamic energy equation. Note that,

$$\rho = \bar{\rho}(z) + \rho'(x, y, z, t),$$

$$p = \bar{p}(z) + p'(x, y, z, t) \text{ and}$$

$$\frac{d}{dt} = \frac{\partial}{\partial t} + u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y} + w\frac{\partial}{\partial z},$$

where $\rho'(x, y, z, t)$ and $p'(x, y, z, t)$ are the deviations of the density and the pressure from the mean density $\bar{\rho}(z)$ and the mean pressure $\bar{p}(z)$, respectively.

Since satellite images do not carry any information about variations of pressure and density, there is no value in incorporating the pressure term or the density term in the optical flow model. On the other hand, brightness changes between two images may arise due to the changes in pressure and density of the fluid. Therefore, there is no effect from the changes in pressure and density to the observed images and hence we assume that p and ρ are constants. Moreover the vertical velocity component w is relatively small compared to the horizontal velocity components u and v, and this leads us to assume $w = 0$. After making these assumptions, the system of equations in Eq.(7.2) becomes

$$\begin{aligned} \frac{du}{dt} - fv &= 0 \\ \frac{dv}{dt} + fu &= 0 \\ \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} &= 0. \end{aligned} \tag{7.3}$$

Using the advective operator, the system of equations in Eq.(7.3) can be rewritten as

$$\begin{aligned} \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} - fv &= 0 \\ \frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + fu &= 0 \\ \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} &= 0. \end{aligned} \tag{7.4}$$

This system is often called the "Barotrophic" model for incompressible fluids. The computation of the time derivatives of velocity components are not possible when we compute the velocity field between two images. On the other hand, we can assume

that there is no acceleration between each pair of images. Therefore, setting

$$\frac{\partial u}{\partial t} = \frac{\partial v}{\partial t} = 0, \tag{7.5}$$

the system of equations in Eq.(7.9) becomes

$$\begin{aligned} u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} - fv &= 0 \\ u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + fu &= 0 \\ \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} &= 0. \end{aligned} \tag{7.6}$$

Since these equations are valid in incompressible fluids, there exists a stream function $\psi(x, y)$ that represents the governing vector fields of the fluid motion at each time instance. Therefore, replacing the velocity components

$$\langle u, v \rangle = \langle -\psi_y, \psi_x \rangle$$

in the system of equations in Eq.(7.6), a new set of equations can be obtained as

$$\begin{aligned} \psi_y\psi_{yx} - \psi_x\psi_{yy} - f\psi_x &= 0 \\ -\psi_y\psi_{xx} + \psi_x\psi_{xy} - f\psi_y &= 0 \\ -\psi_{yx} + \psi_{xy} &= 0, \end{aligned} \tag{7.7}$$

where $-\psi_{yx} + \psi_{xy} = 0$ is redundant. The next step is to include the terms from the quasi-static equations in an optical flow energy functional.

7.3 Quasi-Static Optical Flow Model

Now we develop a functional to emphasize the coriolis force in the vector field computation. According to the results in Chapter 3 and Chapter 6, the optical flow energy functional with conservation of intensity and the smoothness regularization term in stream function formulation performs well in capturing the local structures in the fluid motions. Therefore, we use the corresponding energy functional as the preliminary stage of the construction of the quasi-static optical flow algorithm. Recall that the stream function formulation of the optical flow energy functional with the conservation of intensity data term and the smoothness regularization term is

$$E(\psi) = \int_\Omega \left(I_t - I_x\psi_y + I_y\psi_x\right)^2 \, d\Omega + \alpha \int_\Omega (\psi_{xx}^2 + \psi_{yy}^2 + \psi_{xy}^2 + \psi_{yx}^2) d\Omega. \tag{7.8}$$

We now obtain an expression which is a functional to represent the errors in both terms in Eq.(7.7) over the domain Ω. For the sake of calculations, we use the L^2 norm to establish the error functional. Then the resulting functional is obtained as

$$QG(\psi) = \int_\Omega \left(\psi_y\psi_{yx} - \psi_x\psi_{yy} - f\psi_x\right)^2 \, d\Omega + \int_\Omega (-\psi_y\psi_{xx} + \psi_x\psi_{xy} - f\psi_y)^2 \, d\Omega. \tag{7.9}$$

In the next step, we include the resulting functional in the energy functional in Eq.(7.8) to develop a new energy functional which emphasizes the coriolis force in the computation of vector fields. The combined functional is obtained by adding the functional in Eq.(7.9) to the functional in Eq.(7.8) with a weighting factor γ.

$$\begin{aligned} E(\psi) = & \int_\Omega \left(I_t - I_x\psi_y + I_y\psi_x\right)^2 \, d\Omega + \alpha \int_\Omega (\psi_{xx}^2 + \psi_{yy}^2 + \psi_{xy}^2 + \psi_{yx}^2) d\Omega \\ & + \gamma \int_\Omega \left(\psi_y\psi_{yx} - \psi_x\psi_{yy} - f\psi_x\right)^2 \, d\Omega \\ & + \gamma \int_\Omega (-\psi_y\psi_{xx} + \psi_x\psi_{xy} - f\psi_y)^2 \, d\Omega \end{aligned} \tag{7.10}$$

The next step is to compute the Euler-Lagrange equation for the functional in Eq.(7.10). We obtain the complete Euler-Lagrange equation by taking the sum of the gradients of each part of the functional. As we already know, the gradients of the functionals

$$\int_\Omega \left(I_t - I_x\psi_y + I_y\psi_x\right)^2 \, d\Omega \text{ and } \int_\Omega \left(\psi_{xx}^2 + \psi_{yy}^2 + \psi_{xy}^2 + \psi_{yx}^2\right) d\Omega \quad \text{are}$$

$$2A^*(I_t + A)\psi \quad \text{and} \quad (B + B^*)\psi$$

respectively, where A and B are obtained as

$$A = -I_x D_y + I_y D_x \quad \text{and} \quad B = D_{xx}D_{xx}^* + D_{yy}D_{yy}^* + D_{xy}D_{xy}^* + D_{yx}D_{yx}^*.$$

Here $D_{\bullet\bullet}$ are the operators as we introduced in Sec. 3.1.1 to compute the partial derivatives of a given array with respect to $\bullet\bullet$. To complete the Euler-Lagrange equation to the functional in Eq.(7.10), we only have to find the gradient of the quasi-static terms. We compute the gradient components of those two pieces separately.

7.3.1 Quasi-Static Euler-Lagrange Equations

In the computation of Euler-Lagrange equations for the functional in Eq.(7.10) which comes from the quasi-static equations, first we consider the functional

$$QG1(\psi) = \int_\Omega \left(\psi_y\psi_{yx} - \psi_x\psi_{yy} - f\psi_x\right)^2 \, d\Omega \tag{7.11}$$

to determine the gradient. Now to compute the gradient of the functional in Eq.(7.11), we add an increment Φ to the stream function ψ with a parameter τ. Then the

resulting functional is obtained as

$$QG1(\psi+\tau\Phi) = \int_\Omega [(\psi_y+\tau\Phi_y)(\psi_{yx}+\tau\Phi_{yx}) \\ -(\psi_x+\tau\Phi_x)(\psi_{yy}+\tau\Phi_{yy}) - f(\psi_x+\tau\Phi_x)]^2 \, d\Omega \tag{7.12}$$

Taking the derivative of the functional in Eq.(7.12) with respect to τ and evaluating the resulting derivative at $\tau = 0$, we have

$$\frac{d}{d\tau} QG1(\psi+\tau\Phi)\Big|_{\tau=0} = 2\int_\Omega (\psi_y\psi_{yx} - \psi_x\psi_{yy} - f\psi_x) \\ (\psi_y\Phi_{yx} + \Phi_y\psi_{yx} - \psi_x\Phi_{yy} - \Phi_x\psi_{yy} - f\Phi_x)\, d\Omega. \tag{7.13}$$

Let $QG1' = \frac{d}{d\tau} QG1(\psi+\tau\Phi)\Big|_{\tau=0}$. Then further simplification of the Eq.(7.13) yields

$$\begin{aligned} QG1' = 2\int_\Omega \Big[&\psi_y\psi_{yx}\psi_y\Phi_{yx} + \psi_y\psi_{yx}\psi_{yx}\Phi_y - \psi_y\psi_{yx}\psi_x\Phi_{yy} \\ &- \psi_y\psi_{yx}\psi_{yy}\Phi_x - f\psi_y\psi_{yx}\Phi_x - \psi_x\psi_{yy}\psi_y\Phi_{yx} \\ &- \psi_x\psi_{yy}\psi_{yx}\Phi_y + \psi_x\psi_{yy}\psi_x\Phi_{yy} + \psi_x\psi_{yy}\psi_{yy}\Phi_x \\ &- f(-\psi_x\psi_{yy}\Phi_x + \psi_x\psi_y\Phi_{yx} + \psi_x\psi_{yx}\Phi_y \\ &- \psi_x\psi_x\Phi_{yy} - \psi_x\psi_{yy}\Phi_x - f\psi_x\Phi_x)\Big]\, d\Omega. \end{aligned} \tag{7.14}$$

Now the gradient $GQG1(\psi)$ of the functional which we obtained in Eq.(7.14) can be simplified as

$$\begin{aligned} GQG1(\psi) = 2[&D^*_{yx}\psi_y\psi_{yx}\psi_y + D^*_y\psi_y\psi_{yx}\psi_{yx} - D^*_{yy}\psi_y\psi_{yx}\psi_x \\ &- D^*_x\psi_y\psi_{yx}\psi_{yy} - fD^*_x\psi_y\psi_{yx} - D^*_{xy}\psi_x\psi_{yy}\psi_y \\ &- D^*_y\psi_x\psi_{yy}\psi_{yx} + D^*_{yy}\psi_x\psi_{yy}\psi_x + D^*_x\psi_x\psi_{yy}\psi_{yy} \\ &- f(-D^*_x\psi_x\psi_{yy} + D^*_{yx}\psi_x\psi_y + D^*_y\psi_x\psi_{yx} \\ &- D^*_{yy}\psi_x\psi_x - D^*_x\psi_x\psi_{yy} - fD^*_x\psi_x)], \end{aligned} \tag{7.15}$$

using the derivative operators. Furthermore, we can consider the second part of the quasi-static functional in Eq (7.10) and we can write that as

$$QG2(\psi) = \int_\Omega \left(-\psi_y\psi_{xx} + \psi_x\psi_{xy} - f\psi_y\right)^2 \, d\Omega. \tag{7.16}$$

As we did in Eq.(7.12) for the first part of the functional in Eq.(7.11) to determine the gradient, we can add an increment Φ with a parameter τ to the functional and then take the derivative of the functional with respect to τ. Then, as in Eq.(7.14), we evaluated the resulting functional at $\tau = 0$ and the following functional

$$\begin{aligned} \frac{d}{d\tau}QG2(\psi + \tau\Phi)\Big|_{\tau=0} = 2\int_\Omega &(-\psi_y\psi_{xx} + \psi_x\psi_{xy} - f\psi_y) \\ &(-\psi_y\Phi_{xx} - \Phi_y\psi_{xx} + \psi_x\Phi_{xy} + \Phi_x\psi_{xy} - f\Phi_y)\, d\Omega \end{aligned} \tag{7.17}$$

was obtained. After simplifying the integral in Eq.(7.17), we have

$$\begin{aligned} GQG2(\psi) = 2\int_{\Omega}[&\psi_y\psi_{xx}\psi_y\Phi_{xx} + \psi_y\psi_{xx}\psi_{xx}\Phi_y - \psi_y\psi_{xy}\psi_x\Phi_{xy} \\ &- \psi_y\psi_{xx}\psi_{xy}\Phi_x + f\psi_y\psi_{xx}\Phi_y - \psi_x\psi_{xy}\psi_y\Phi_{xx} \\ &- \psi_x\psi_{xy}\psi_{xx}\Phi_y + \psi_x\psi_{xy}\psi_x\Phi_{xy} + \psi_x\psi_{xy}\psi_{xy}\Phi_x \\ &+ f(-\psi_x\psi_{xy}\Phi_y + \psi_y\psi_y\Phi_{xx} + \psi_y\psi_{xx}\Phi_y \\ &- \psi_y\psi_x\Phi_{xy} - \psi_y\psi_{xy}\Phi_x^+ f\psi_y\Phi_y)]\, d\Omega. \end{aligned} \tag{7.18}$$

Simplifying and then substituting the derivative operators, the gradient $GQG2(\psi)$ of the functional in Eq.(7.16) becomes

$$\begin{aligned} GQG2(\psi) = 2[&D_{xx}^*\psi_y\psi_{xx}\psi_y + D_y^*\psi_y\psi_{xx}\psi_{xx} - D_{xy}^*\psi_y\psi_{xy}\psi_x \\ &- D_x^*\psi_y\psi_{xx}\psi_{xy} + fD_y^*\psi_y\psi_{xx} - D_{xx}^*\psi_x\psi_{xy}\psi_y \\ &- D_y^*\psi_x\psi_{xy}\psi_{xx} + D_{xy}^*\psi_x\psi_{xy}\psi_x + D_x^*\psi_x\psi_{xy}\psi_{xy} \\ &+ f(-D_y^*\psi_x\psi_{xy} + D_{xx}^*\psi_y\psi_y + D_y^*\psi_y\psi_{xx} \\ &- D_{xy}^*\psi_y\psi_x - D_x^*\psi_y\psi_{xy} + fD_y^*\psi_y)]. \end{aligned} \tag{7.19}$$

We now have the gradients or the Euler-Lagrange equations for both separate functional terms of the quasi-static energy functional in Eq.(7.9). Therefore, the complete Euler-Lagrange equation for the quasi-static energy functional in Eq.(7.9) can be written as

$$GQG(\psi) = \left[GQG1(\psi) + GQG2(\psi)\right]/2, \tag{7.20}$$

where we divide the right hand side by 2 for the sake of simplicity. Hence the Euler-Lagrange equation for the energy functional in Eq.(7.10) is written as

$$2A^*(I_t + A\psi) + \alpha(B^* + B)\psi + 2\gamma\, GQG(\psi) = 0. \tag{7.21}$$

Replacing α by 2α, the Euler-Lagrange equation can be written as

$$A^*(I_t + A\psi) + \alpha(B^* + B)\psi + \gamma\, GQG(\psi) = 0. \tag{7.22}$$

Now we assume that a minimizer for the energy functional in Eq.(7.10) exists; hence we develop a gradient descent iterative scheme to determine the optimal stream function ψ. For a given initial stream function ψ^0, the gradient descent approach is

$$\psi^{k+1} = \psi^k - \Delta\tau \left[A^*(I_t + A\psi^k) + \alpha(B^* + B)\psi^k + \gamma\, GQG(\psi^k)\right], \tag{7.23}$$

where $\Delta\tau$ is the step size corresponding to each iteration and k is the iteration number. The stopping criteria for the gradient descent method depends on both the maximum number of iterations k_{max} and the threshold (Δ) value for the errors of the stream function in successive iterates. We stop the algorithm when k reaches k_{max} or the error of stream function in successive iterates is smaller than Δ.

For the initial ψ^0, usually we use the null stream function

$$\psi^0\,(x, y) = 0.$$

However, there are some instances where the gradient descent method does not converge for some initial conditions including the null stream function. In this case, first we have to identify a suitable initial condition for the gradient descent method. We achieve this by introducing a fixed point method by rearranging the terms in the

Euler-Lagrange equation in Eq.(7.22) as

$$[A^*A + \alpha(B^* + B)]\,\psi = -A^*I_t - \gamma\, GQG(\psi). \tag{7.24}$$

Now for a given initial stream function ψ^0, the solution is obtained by the following iterative procedure with appropriate parameter values for α and γ

$$\psi^{m+1} = -\left[A^*A + \alpha(B^* + B)\right]^{-1}\left[A^*I_t + \gamma\, GQG(\psi^m)\right], \tag{7.25}$$

where m is the iteration number. We use few iterations and let $\psi^\bullet$ be a solution after the required number of iterations. Then we apply the gradient descent method with the initial stream function $\psi^\bullet$ to reach the optimal solution for the quasi-static optical flow algorithm. This combined method is called a homotopy scheme. Note that, in fixed point method, we only use few iterations as the magnitude of the points in the stream function increases with the number of iterations.

Now that the numerical scheme is established by combining fixed point initial solution with gradient descent method, we want to check the accuracy. Therefore, we introduce a benchmark data set in Sec. 7.4 to which we employ the quasi-static optical flow method, before we address the real data.

7.4 Benchmark Data Set

In this section, we construct a benchmark data set to apply the algorithm we developed in Sec. 7.3. In the construction, we use the gyre flow as shown in image (c) in Fig. 2.4, on a rotating coordinate system. Recall that the gyre flow is given by

$$\langle u, v\rangle = \langle -\pi \sin(\pi x)\cos(\pi y), \pi\cos(\pi x)\sin(\pi y)\rangle, \tag{7.26}$$

where $(x, y) \in [0, 1] \times [0, 1]$. Letting θ be the angle after time t, then the new coordinates (x', y') become

$$\begin{aligned} x' &= x\cos\theta + y\sin\theta \\ y' &= -x\sin\theta + y\cos\theta, \end{aligned}$$

where $\theta = \omega t$. For this data set, the rotation rate ω is chosen as $\omega = \pi/10\ rad/s$. Now the resulting gyre flow on the rotating coordinate system is obtained as

$$\langle u', v' \rangle = \langle -\pi \sin(\pi x')\cos(\pi y'), \pi\cos(\pi x')\sin(\pi y') \rangle. \tag{7.27}$$

Next we evolve an initial density using the velocity components found in Eq.(7.27) according to the continuity equation in Eq.(2.29) with $dt = 0.1$. We select two images after 5 seconds to apply the quasi-static optical flow algorithm. Image (a) in Fig. 7.2 shows the initial velocity field of the gyre that is represented in Eq.(7.26). Image (b) shows the flow field after 5 seconds in rotating the coordinates. Images (c) and (d) in Fig. 7.2 are the preceding and the proceeding images of the vector field on image (c).

We now apply the quasi-static method on images (c) and (d) in Fig. 7.2. The quasi-gestrophic algorithm produces the gyre as we expect with the MAE 2.962° for the parameters $\alpha = 1e-2$ and $\gamma = 0.09$. The resulting flow field is shown in image (a) in Fig. 7.3. We can also apply the other optical flow algorithms assuming that there is no coriolis effect on images (c) and (d) in Fig. 7.2. The conservation of intensity data term and the smoothness regularization term in u-v formulation produced the best result with the minimum MAE 3.704°. The resulting flow field is shown in the image (b) in Fig. 7.3.

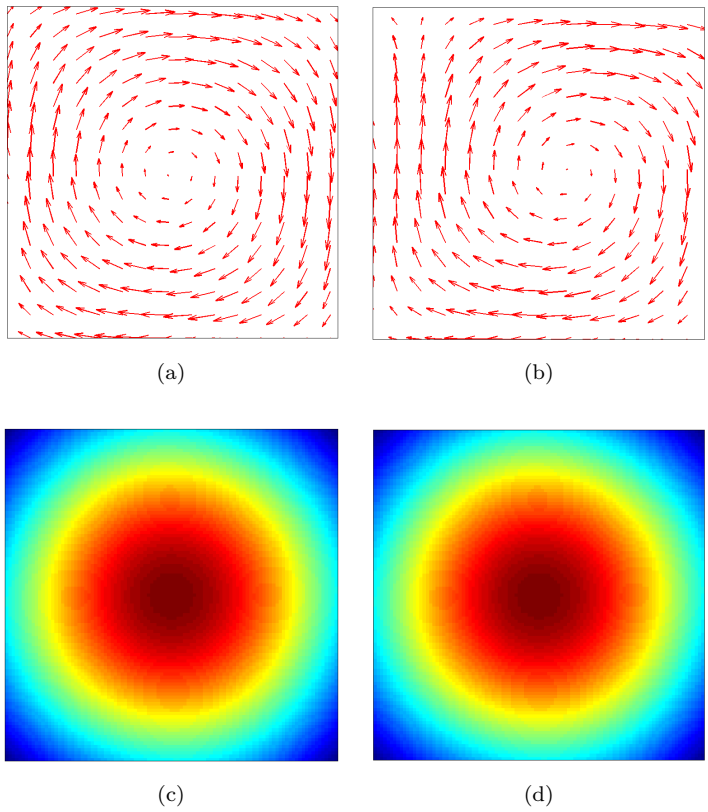
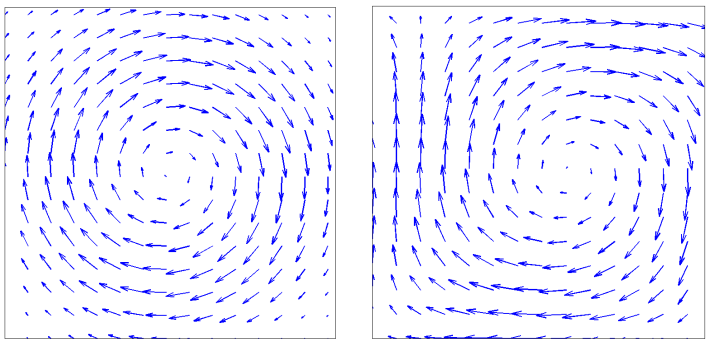


Figure 7.2: Gyre on Rotating coordinates – The images (a) and (b) show the initial flow field and a selected flow field after 5 seconds on the rotating coordinates respectively. An initial density on the flow in Eq.(7.27) evolved according to Eq.(2.29). Images (d) represents the evolution of image (c) under the flow field in image (b).

The results from the quasi-static method and the conservation of intensity data term with the smoothness regularization term in the u-v formulation show the effect of the coriolis force on images. The quasi-static method provides an appropriate correction on the effect of the coriolis force. As a real application for the quasi-static optical flow method, we now introduce a data set which represents Jupiter's atmosphere.



(a) (b)

Figure 7.3: Computed flow on Rotating gyre – The first image shows the computed flow field from the quasi-static algorithm on images (c) and (d) in Fig. 7.2. The true flow is shown in image (a) of Fig. 7.2. The second image shows the computed flow field from the conservation of intensity data term and the smoothness regularization term in u-v formulation. The true flow field is shown in image (b) of Fig. 7.2.

7.5 Jupiter

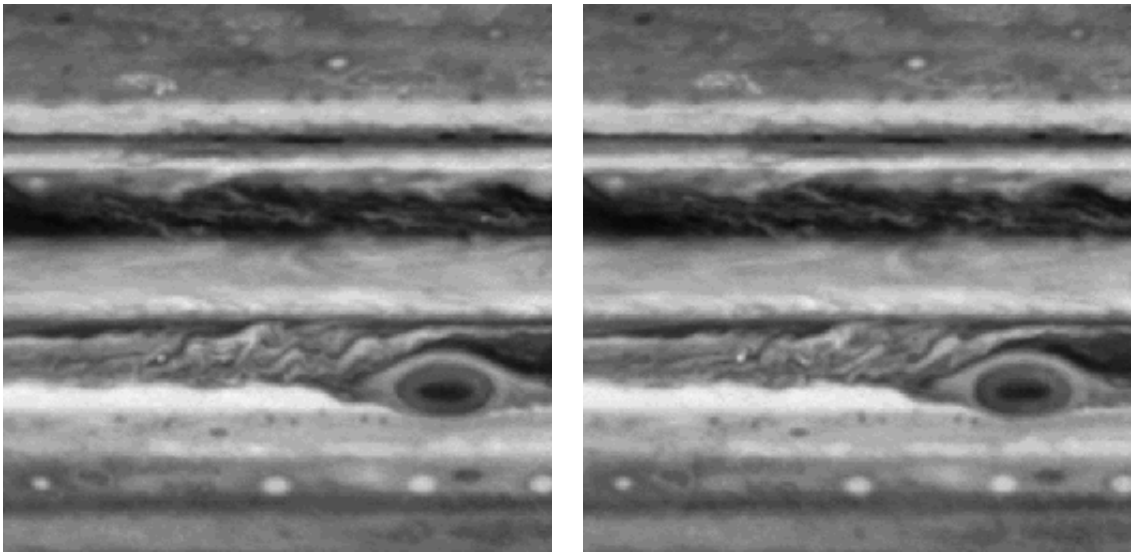
In this section, we test our algorithms on a real sequence of satellite images that represents the motion of Jupiter's atmosphere. Unlike the regular optical flow methods, the quasi-static method requires more information about the system other the image data. Therefore, we first discuss the planet Jupiter and its interesting features before getting into the flow computations from satellite images. Jupiter is the largest planet in our solar system and it is the fifth from the Sun. Furthermore, Jupiter has the fastest rotation rate in our solar system; it completes one day in 9.92 earth hours which is called a Jovian day. Another interesting fact is that Jupiter does not have a solid surface and its atmosphere mostly consists of hydrogen and helium [99]. Other than the reasons mentioned above, compared to the other planets, Jupiter is special because it has a giant [100] red vortex called the Great Red Spot (GRS) whose diameter is three times the diameter of the earth. The GRS is located just below the equator. It is an oval shape violent storm that rotates counterclockwise about its center. The apparent color of the GRS may be due to small amounts of sulfur and

phosphorus in the ammonia crystals in Jupiter's clouds [100]. It has remained almost the same more for than 300 years.

Some recent work related to Jupiter's atmospheric motions [3, 5, 19, 101] motivates us to compute vector field in an analysis of atmospheric motion near the GRS. When reviewing the relevant work in [101], we uncovered that the author developed governing equations for Jupiter's atmospheric motion and then solved the equations numerically to obtain the velocity fields. However, the resulting velocity fields are compared with the existing mean velocities. The authors in [3,5] discussed the transport barriers in the planetary atmosphere including Jupiter and in this work, they used a potential vorticity gradient to obtain the results. In [19], the authors address the change of size in GRS and Oval BA, which is another oval shaped spot located just below the GRS, from the year 1996 to 2006. In this case the authors obtained the vector fields from Advected Correction Correlation Image Velocimetry (ACCIV) method.

In this section we introduce a data set on Jupiter's atmospheric motion available at [102]. Our motivation is to use the time dependent velocity fields to analyze the transport barriers in Jupiter's atmosphere near the GRS. We use the quasi-static optical flow method to compute the velocity fields as the data was affected by the coriolis force. This complete data set was taken by NASA's Cassini spacecraft during the period of 7 Jovian days from October, 1, 2000 to October, 5, 2000. In this data set, the counter-clockwise atmospheric motion around the GRS is clearly visible. The first two images of the data set are shown in Fig. 7.4.

The images ranged from 50 degrees south to 50 degrees north with respect to the equator and 100 degrees from east to west. The exact longitudes are not available. However, to apply the quasi-static method, the exact latitude values are sufficient. Two images in the Fig. 7.4 are gray scale and when we can represent the images

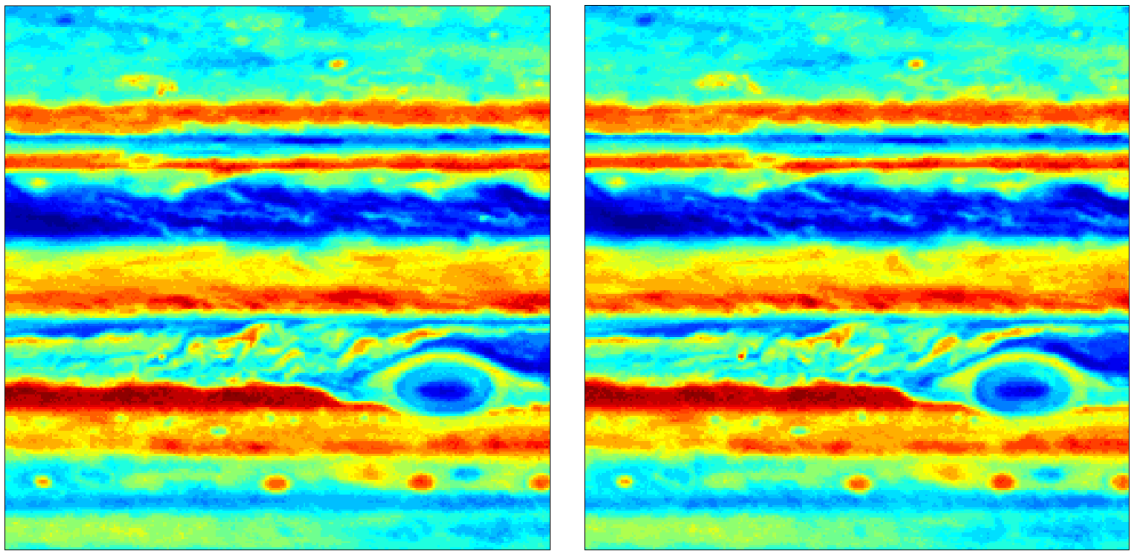


(a) image 1 (b) image 2

Figure 7.4: Jupiter Images –Images (a) and (b) show two consecutive observations of Jupiter's atmospheric motion that are one Jovian day apart. The complete data set consists of 7 images and was obtained by NASA's Cassini spacecraft from October, 1, 2000 to October, 5, 2000 [102]. The latitudes of these images ranged from 50 degrees south to 50 degrees north and longitude expands 100 degrees from east to west.

in Matlab with default color scale. Fig. 7.5 shows the Matlab representation of two images in Fig. 7.4, where each pixel represents the corresponding intensity value which varies from 0 to 255..

When we developed the optical flow algorithm, we assumed that the motion between two images is small. However, the motion between two images in the Fig. 7.5 is large (more than one pixel) and hence we use cubic spline to make the motion be small by including 4 artificial frames in between two actual frames. Also to avoid having the same index in nearby points, we smooth the images spatially using Gaussian smoothing with the standard deviation of 0.04. The images (a) and (b) in Fig. 7.6 show the first two images after smoothing in both the time and the spatial directions.



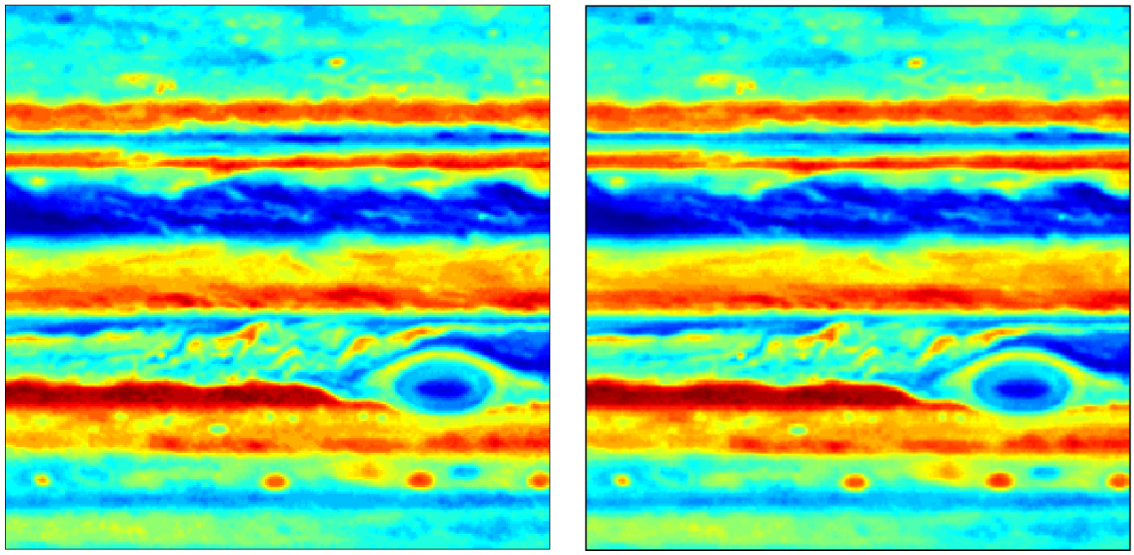
(a) color image 1

(b) color image 2

Figure 7.5: Jupiter Images in Matlab – Images (a) and (b) show the appearance of two consecutive gray images shown in Fig. 7.4 in Matlab according to the default color scale. The color of each pixel represents the image intensity and it varies from 0 to 255. The same color scale is used for the rest of this work for the sake of clear visualization

As we explained earlier, our motivation is to provide velocity fields to analyze the atmospheric motion around the GRS. Therefore, we first compute the vector fields near the GRS. Fig. 7.7 shows a rectangular selected area around the GRS on the image (a) in Fig. 7.6. The selected region was coarse and therefore, we increased the resolution of the area by a factor two to get spatially smoothed images.

Images (a) and (b) in Fig. 7.8 show a selected area of images (a) and (b) in Fig. 7.6. However these two cropped images represent the selected area after increasing the spatial resolution by two. In forthcoming calculations of vector fields, this selected area around the GRS will be used.



(a)Smoothed image 1

(b) Smoothed image 2

Figure 7.6: Jupiter smooth images – Images (a) and (b) show the first two consecutive images after smoothing seven images in the data set in both time and spatial direction. When we smooth in time direction, we include four artificial images in between two actual images. Therefore, image (a) represents the same image in Fig. 7.5 despite the spatial smoothing. However the image (b) represents an artificial image just after small motion of image (a).

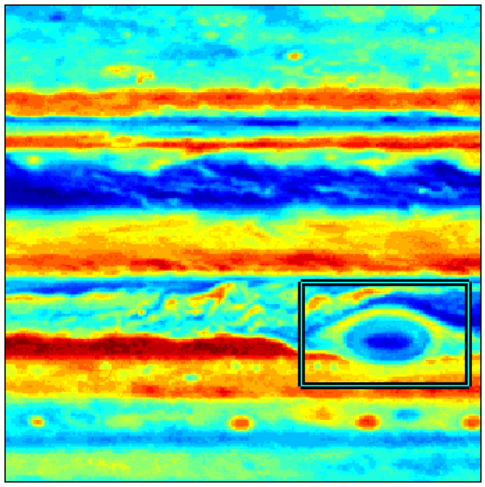
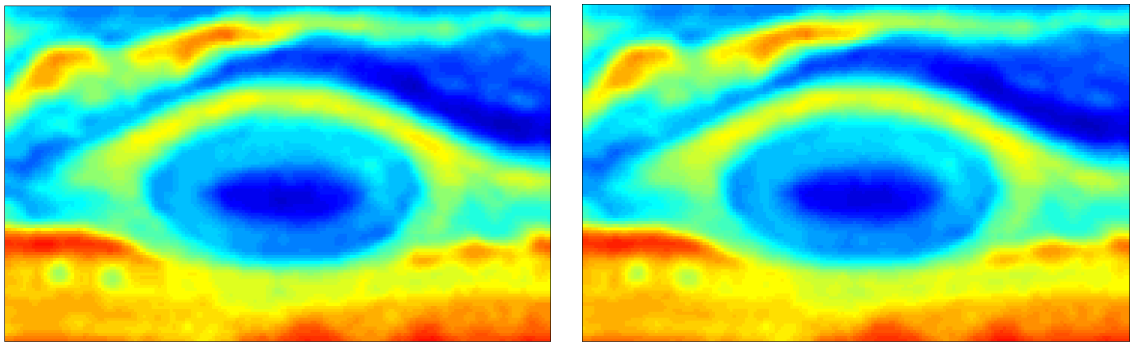


Figure 7.7: The GRS – The image shows the rectangular area selected on image (a) in Fig. 7.6. This rectangular area contains the GRS whose dynamics of atmospheric motion we are interested in analyzing.



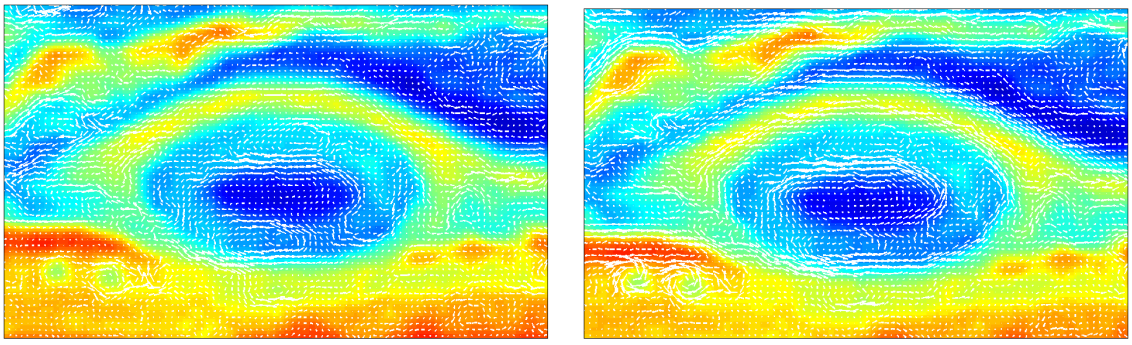
(a) Cropped image 1 (b) Cropped image 2

Figure 7.8: GRS Images –Images (a) and (b) show the selected area around the GRS on the images (a) and (b) in the Fig. 7.8. The spatial resolution of these cropped images has been increased by two.

In the next step, we apply the stream function optical flow method and the quasi-geostropic optical flow algorithm to the images shown in Fig. 7.8. The vector field from the stream function method captures a reasonable solution to the boundary of the GRS, but the flow inside is not accurate. On the other hand, the quasi-static method reconstructs a reasonable solution to the GRS for both the boundaries and the interior. The solutions from the stream function method and the quasi-static method are shown in images (a) and (b) of Fig. 7.9, respectively.

the computed flow field from both methods capture upper part of the GRS flow accurately. Also, the interior flow in the upper half of the GRS are reconstructed well. However, the quasi-static method captures the lower part of the GRS flow over the stream function method.

One way of extending the quasi-static method is to introduce different data fidelities and different regularization terms to combine with quasi-static energy functional. Our interest, however, is to extend this method towards the multi-time step method as we expect to apply this algorithm on images of time dependent systems. Therefore, in Sec. 7.6, we discuss the quasi-static Multi-Time Step Method.



(a) stream function flow (b) quasi-static flow

Figure 7.9: Computed flow on GRS – Image (a) shows the computed velocity field for the images shown in Fig. 7.8 using stream function method. Image (b) shows the computed vector field from the quasi-static method. Both solutions are reasonable for the boundaries of the GRS. The quasi-static method tries to produce a vortex flow inside the GRS as the expected flow field shown in [19].

7.6 Quasi-Static Multi-Time Step Method

We now apply the a multi-times Step method as introduced in Chapter 6 for the quasi-static optical flow approach using the energy functional we developed in Eq.(7.10). Multi-time step method allows us to impose regularity in time direction and it provides a continuity of the flow in time direction. We explained the construction of multi-time step method in Chapter 6, and we are going to follow the same steps to derive the multi-time step method for quasi-static formulation. Before writing the general version of the multi-time step method, we consider the step size $n = 2$ and develop the energy functional. Since we are going to compute two stream functions ψ_1 and ψ_2 simultaneously, we first obtain two energy functionals corresponding to ψ_1 and ψ_2 using the Eq.(7.10) and then add them together. Next we add the functional in Eq.(6.3) to the functional we obtained in the previous step with a weighting pa-

rameter β to emphasize the close relationship between two the stream functions. The resulting energy functional to be minimized becomes

$$\begin{aligned}
E(\psi_1,\psi_2) = &\int_\Omega \left(I_{1t} - I_{1x}\psi_{1y} + I_{1y}\psi_{1x}\right)^2 \, d\Omega + \alpha \int_\Omega (\psi_{1xx}^2 + \psi_{1yy}^2 + \psi_{1xy}^2 + \psi_{1yx}^2) d\Omega \\
&+ \gamma \int_\Omega \left(\psi_{1y}\psi_{1yx} - \psi_{1x}\psi_{1yy} - f\psi_{1x}\right)^2 \, d\Omega \\
&+ \gamma \int_\Omega (-\psi_{1y}\psi_{1xx} + \psi_{1x}\psi_{1xy} - f\psi_{1y})^2 \, d\Omega \\
&+ \beta \int_\Omega \left(\psi_1 - \psi_2\right)^2 \, d\Omega + \int_\Omega \left(I_{2t} - I_{2x}\psi_{2y} + I_{2y}\psi_{2x}\right)^2 \, d\Omega \\
&+ \alpha \int_\Omega (\psi_{2xx}^2 + \psi_{2yy}^2 + \psi_{2xy}^2 + \psi_{2yx}^2) d\Omega \\
&+ \gamma \int_\Omega \left(\psi_{2y}\psi_{2yx} - \psi_{2x}\psi_{2yy} - f\psi_{2x}\right)^2 \, d\Omega \\
&+ \gamma \int_\Omega (-\psi_{2y}\psi_{2xx} + \psi_{2x}\psi_{2xy} - f\psi_{2y})^2 \, d\Omega. \qquad (7.28)
\end{aligned}$$

To determine the optimal ψ_1 and ψ_2, we need at least three images which represent three consecutive time instances of the observed system. Here the subscript '1' denotes all the terms between images 1 and 2 and the subscription '2' denotes all the terms between images 2 and 3. Since there are two functions, ψ_1 and ψ_2 in the functional, there will be two Euler-Lagrange equations to be solved for ψ_1 and ψ_2. Apart from the term $\int_\Omega \left(\psi_1 - \psi_2\right)^2 \, d\Omega$, contributions from the other integrals to the Euler-Lagrange equations can be directly obtained from the Eq.(7.22) by substituting ψ_1 and ψ_2. The new contributions for the Euler-Lagrange equations for ψ_1 and ψ_2 from the $\int_\Omega \left(\psi_1 - \psi_2\right)^2 \, d\Omega$ are $2\left(\psi_1 - \psi_2\right)$ and $-2(\psi_1 - \psi_2)$, respectively. The resulting Euler-Lagrange equations are given by

$$\begin{aligned}
&A_1^*(I_{1t} + A_1\psi_1) + \alpha(B^* + B)\psi_1 + \beta(\psi_1 - \psi_2) + \gamma GQG(\psi_1) = 0 \\
&A_2^*(I_{2t} + A_2\psi_2) + \alpha(B^* + B)\psi_2 - \beta(\psi_1 - \psi_2) + \gamma GQG(\psi_2) = 0 \qquad (7.29)
\end{aligned}$$

As we derived for $n = 1$, we are going to build an iterative method to solve the system. The solution $z = [\psi_1, \psi_2]^T$ is obtained from the gradient descent method by solving the iterative scheme as

$$z^{k+1} = z^k - \Delta\tau w\left(z^k\right), \tag{7.30}$$

where

$$w\left(z^k\right) = \begin{bmatrix} A_1^*(I_{1t} + A_1\psi_1^k) + \alpha(B^* + B)\psi_1^k + \beta(\psi_1^k - \psi_2^k) + \gamma GQG(\psi_1^k) \\ \\ A_2^*(I_{2t} + A_2\psi_2^k) + \alpha(B^* + B)\psi_2^k - \beta(\psi_1^k - \psi_2^k) + \gamma GQG(\psi_2^k) \end{bmatrix}.$$

Stopping criteria is similar to the step size $n = 1$ and the initial condition z^0 is determined as we explained in the case of $n = 1$. To determine a suitable initial condition, we use the fixed point iteration method which we introduced in Eq.(7.25) for $n = 1$. Extending the procedure to $n = 2$, the system to be solved for z^{m+1} is

$$z^{m+1} = K^{-1}b(z^m), \tag{7.31}$$

where

$$z^{m+1} = \begin{bmatrix} \psi_1^{m+1} \\ \\ \psi_2^{m+1} \end{bmatrix}, \quad K = \begin{bmatrix} A_1^*A_1 + \alpha(B^* + B) & 0 \\ \\ 0 & A_2^*A_2 + \alpha(B^* + B) \end{bmatrix}$$

$$\text{and } b(z^m) = \begin{bmatrix} -A_1^*I_{1t} - \beta(\psi_1^m - \psi_2^m) - \gamma GQG(\psi_1^m) \\ \\ -A_2^*I_{2t} + \beta(\psi_1^m - \psi_2^m) - \gamma GQG(\psi_2^m) \end{bmatrix}.$$

As noted in the case of $n = 1$, the number of iterations for the fixed point method is smaller than 10.

Continuing in this manner for any finite number of stream functions, we can generalize the multi-time step method for the quasi-static approach. We now consider computing n stream functions at a time. Then the energy functional corresponding to all the stream functions can be written as

$$\begin{aligned} E(\psi_1, \psi_2, ..., \psi_n) = & \sum_{i=1}^{n} \int_\Omega \left(I_{it} - I_{ix}\psi_{iy} + I_{iy}\psi_{ix}\right)^2 \, d\Omega \\ & + \alpha \sum_{i=1}^{n} \int_\Omega (\psi_{ixx}^2 + \psi_{iyy}^2 + \psi_{ixy}^2 + \psi_{iyx}^2) d\Omega \\ & + \beta \sum_{i=1}^{n-1} \int_\Omega \left(\psi_k - \psi_{k+1}\right)^2 \, d\Omega \\ & + \gamma \sum_{k=i}^{n} \int_\Omega \left(\psi_{iy}\psi_{iyx} - \psi_{ix}\psi_{iyy} - f\psi_{ix}\right)^2 \, d\Omega \\ & + \gamma \sum_{k=i}^{n} \int_\Omega (-\psi_{iy}\psi_{ixx} + \psi_{ix}\psi_{ixy} - f\psi_{iy})^2 \, d\Omega \end{aligned} \tag{7.32}$$

Since there are n stream functions in the functional in Eq.(7.32), we take the first variation, or the gradient, of the functional with respect to each argument variable ψ_s for $s = 1, 2, ...n$. This leads to n Euler-Lagrange equations for the functional in Eq.(7.32) and can be written as

$$A_1^*(I_{1t} + A_1\psi_1) + \beta(\psi_1 - \psi_2) + \alpha(B + B^*)\psi_1 + \gamma GQG(\psi_1) = 0, \text{ for } s = 1$$

$$A_s^*(I_{st} + A_s\psi_s) + \beta(-\psi_{s-1} + 2\psi_s - \psi_{s+1}) + \alpha(B + B^*)\psi_s + \gamma GQG(\psi_s) = 0,$$

for $s = 2, 3, ...n - 1,$

$$A_n^*(I_{nt} + A_n\psi_n) + \beta(\psi_{n-1} - \psi_n) + \alpha(B + B^*)\psi_n + \gamma GQG(\psi_n) = 0, \text{for } s = n,$$

where $A_s = (-I_{sx}D_y + I_{sy}D_x)\,\forall k,\ B = D_{xx}D^*_{xx} + D_{yy}D^*_{yy} + D_{xy}D^*_{xy} + D_{yx}D^*_{yx}$ and $GQG(\psi_s)$ is defined as in the Eq.(7.20). Then we determine the optimal $\psi_1, \psi_2, ..., \psi_n$ from the following gradient descent iterative procedure,

$$z^{k+1} = z^k - \Delta\tau w\left(z^k\right), \tag{7.33}$$

where $w\left(z^k\right)$ is obtained as

$$\begin{bmatrix} A^*_1(I_{1t} + A_1\psi^k_1) + \beta(\psi^k_1 - \psi^k_2) + \alpha(B + B^*)\psi^k_1 + \gamma GQG(\psi^k_1) \\ A^*_2(I_{2t} + A_2\psi^k_2) + \beta(-\psi^k_1 + 2\psi^k_2 - \psi^k_3) + \alpha(B + B^*)\psi^k_2 + \gamma GQG(\psi^k_2) \\ \vdots \\ A^*_{n-1}(I_{(n-1)t} + A_{n-1}\psi^k_{n-1}) + \beta(-\psi^k_{n-2} + 2\psi^k_{n-1} - \psi^k_n) + \alpha(B + B^*)\psi^k_{n-1} + \gamma GQG(\psi^k_{n-1}) \\ A^*_n(I_{nt} + A_n\psi^k_n) + \beta(\psi^k_{n-1} - \psi^k_n) + \alpha(B + B^*)\psi^k_n + \gamma GQG(\psi^k_n) \end{bmatrix}$$

and $z^k = \begin{bmatrix} \psi^k_1 & \psi^k_2 & \dots & \psi^k_n \end{bmatrix}^T$.

Again to determine a suitable initial condition for z^k, we extend the fixed point method in Eq.(7.25) to n stream functions as

$$z^{m+1} = K^{-1}b(z^m), \tag{7.34}$$

where

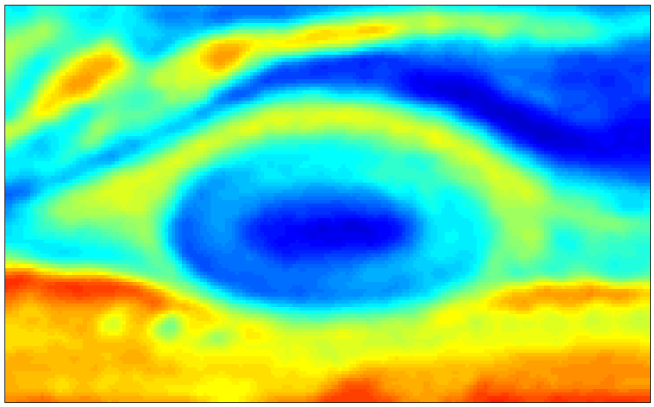
$$K = \begin{bmatrix} A^*_1A_1 + \alpha(B^* + B) & & & & \\ & A^*_2A_2 + \alpha(B^* + B) & & & \\ & & \ddots & & \\ & & & A^*_{n-1}A_{n-1} + \alpha(B^* + B) & \\ & & & & A^*_nA_n + \alpha(B^* + B) \end{bmatrix},$$

$$z^{m+1} = \begin{bmatrix} \psi_1^{m+1} \\ \psi_2^{m+1} \\ \vdots \\ \psi_{n-1}^{m+1} \\ \psi_n^{m+1} \end{bmatrix} \text{ and } b(z^m) = \begin{bmatrix} -A_1^* I_{1t} - \beta(\psi_1^m - \psi_2^m) - \gamma GQG(\psi_1^m) \\ -A_2^* I_{2t} - \beta(-\psi_1^m + 2\psi_2^m - \psi_3^m) - \gamma GQG(\psi_2^m) \\ \vdots \\ -A_{n-1}^* I_{(n-1)t} - \beta(-\psi_{n-2}^m + 2\psi_{n-1}^m - \psi_n^m) - \gamma GQG(\psi_{n-1}^m) \\ -A_n^* I_{nt} + \beta(\psi_{n-1}^m - \psi_n^m) - \gamma GQG(\psi_n^m) \end{bmatrix}.$$

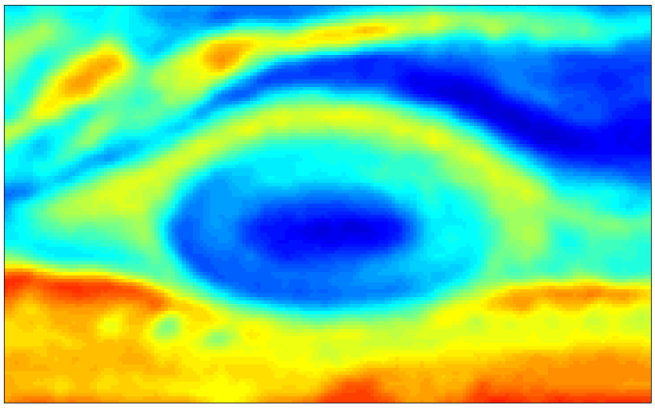
For the fixed point method, we can use any initial condition which produces a reasonable input for the gradient descent method.

Now the quasi-static multi-time step algorithm can be applied to Jupiter's atmosphere which we introduced in Sec. 7.5 and which is available at [102]. Fig. 7.10 shows three consecutive images of the same sequence we presented in Sec. 7.5 and images (a) - (c) show 10th, 11th and 12th images of the sequence.

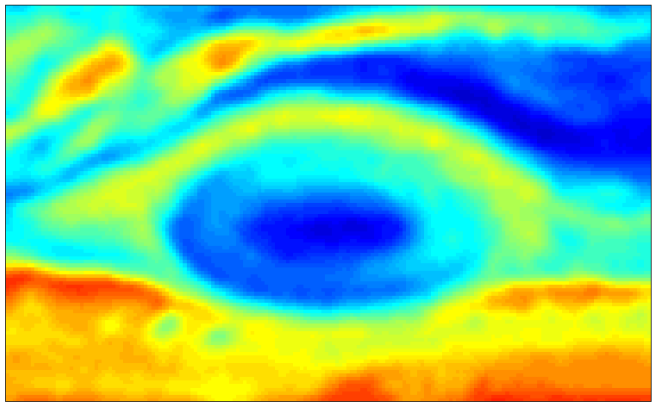
Images (a) - (c) in Fig. 7.11 show the computed flow fields from the quasi-static method with the step size $n = 1$, 2 and 3, respectively on the image (a) in Fig. 7.11. In these computations, we selected the weighting factor γ for the quasi-static term to be 0.092 and the regularization parameters for $n = 1, 2$ and 3 are $\alpha = 10^{-6}, 3 \times 10^{-6}$ and 8×10^{-6}, respectively. In each case, the algorithm was able to capture the vortex flow very accurately. However, when the step size increases, the flows near the boundary of the vortex (GRS) improve.



(a) image 10



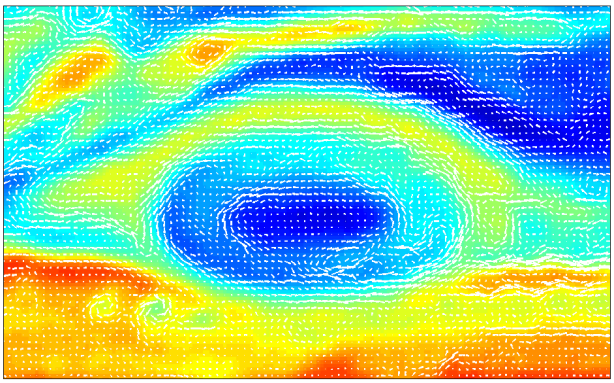
(b) image 11



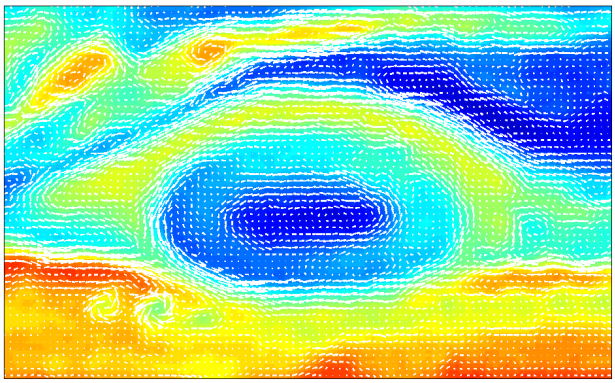
(c) image 12

Figure 7.10: GRS image sequence – Images (a) - (c) show 10 - 12 cropped images from the initial images are shown in Fig. 7.8. The complete data set is available at [102].

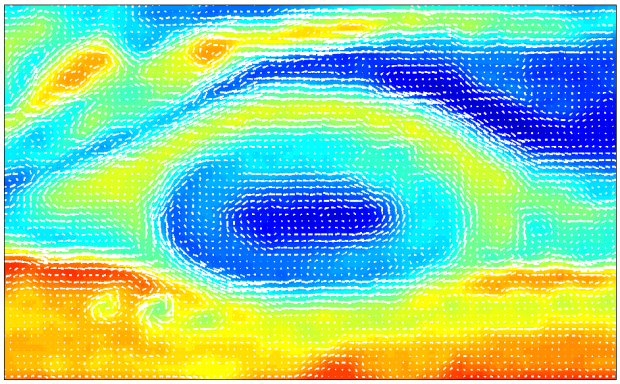
To check accuracy of the computed velocity fields near the GRS from the quasi-static algorithm, we can apply the computed flow fields to check one of the interesting features of the GRS. This interesting feature is that the points inside the GRS rarely leave the GRS. First we created a handmade boundary for the GRS. Image (a) in Fig. 7.12 shows the GRS boundary in black and the points inside the GRS are red whereas the points outside the GRS are green. Image (a) represents the initial position. Image (b) shows the positions after 25 time units when we integrate the positions in image (a) using the computed velocity from the quasi-static multi-time step method $n = 2$. Apart from a few points, it can be observed that the points on the inside and on the outside do not cross the boundary. This result validates the accuracy of the quasi-static method.



(a) flow with $n = 1$

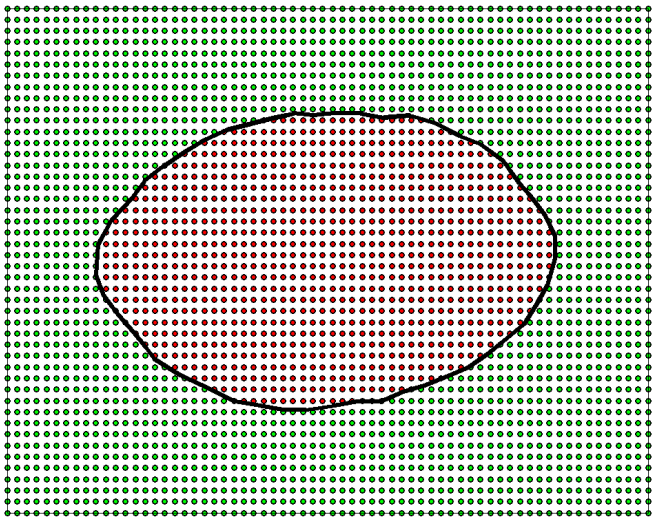


(b) flow with $n = 2$

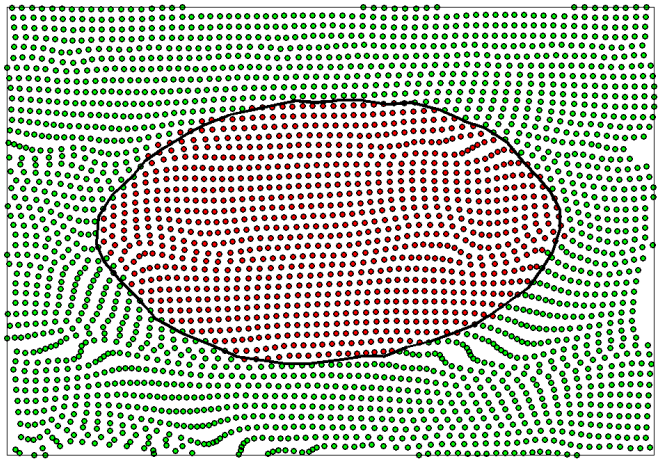


(c) flow with $n = 3$

Figure 7.11: Quasi-Static multi-time step flow - images (a) - (c) shows the computed flow from the quasi-static multi-time step method on the image (a) in Fig. 7.10 with step sizes $n = 1, 2$ and 3, respectively. Resulting flow fields from each step size are reasonable, especially around the GRS. It is clearly visible that the vortex structure improves with the step size.



(a) initial density



(b) density after 25 images

Figure 7.12: GRS particle advection – Images (a) shows points inside GRS and points outside the GRS using a handmade boundary. We integrated the points in image (a) using the computed flow from the quasi-static multi-time step method with $n = 2$. Positions after 25 images are shown in image (b). Other than a few points, inside and outside points do not cross the boundary.

Chapter 8

Lagrangian Coherent Structures

In the first seven chapters, we discussed the computation of vector fields using an optical flow approach. Then we extended the method for various directions to adopt the Horn and Schunck optical flow algorithm for real world applications. After all these implementations, we employ the computed flow fields of an observed system to analyze the dynamics of the observed system. We now present the important role the velocity field plays in analysis of dynamics in fluid systems.

Scientists are interested in analyzing systems not only on the earth, but also on the other planets. In such scientific analysis, scientists employ various techniques to analyze systems. Since we are interested in fluid systems, we can narrow our discussion to the analysis of fluid systems. Some popular approaches used in analyzing fluid systems are developed by applying the transfer operator method [1,103–106] and determining Lagrangian coherent structures [1], and coherent sets [8].

Determining the transport and mixing barriers is an important part of the study of fluid systems such as ocean currents, heat waves, and cloud movements. These transport and mixing barriers help us to analyze the system as well as to predict the system behavior. In many cases, Lagrangian Coherent Structures (LCSs) are employed to determine the transport and mixing of fluid systems.

Definition 7. *[7] Lagrangian Coherent Structures are the ridges in the FTLE field. Ridges are special gradient lines of the FTLE field that are transverse to the direction of minimum curvature.*

In some recent work, LCSs are used to identify mesoscale oceanic eddies from surface ocean currents [2] and to predict how to affect the changes in flow to the transport and mixing [107]. Common approaches use to determine Lagrangian coherent structures of fluid systems are to compute Finite Time Lyaponov Exponents (FTLE) [6, 108, 109] or Finite Size Lyaponov Exponents (FSLE) [110] and then extract the LCSs from the computed FTLE or FSLE field.

In this chapter we determine LCSs to analyze transport and mixing of fluid systems based on Finite Time Lyapunov Exponents. The extracted LCSs provide the skeleton of pseudo-barriers to transport and mixing in dynamical systems and they are of co-dimension one relative to the dimension of the considered system. In other words, LCSs separate the dynamically distinct regions in a dynamical system. These structures are not visible and are hard to see directly from the vector fields or by evolving the trajectories [7, 111]. The flux of particles through the LCSs are zero or close to zero [7, 112] with sufficiently large integration time for the computation of FTLE. Since the computation of FTLE requires velocity fields governing the system which we are interested in, as an application of the computed vector fields, we will discuss a transport analysis inferred directly from observed spatio-temporal movie data of a particular system. Next we introduce the numerical computation of finite time Lyapunov exponents for given velocity data in Sec. 8.1.

8.1 Mixing and Transport Barriers

Toward the identification of mixing and transport barriers [1, 113–115], we compute finite time Lyapunov exponents which are scalar values for each point in the domain

D of the system as explained in [6, 108]. Then we extract LCSs from the computed FTLE field on D and the flux across these structures are almost zero [7, 112].

A practical approach of computing FTLE of a time-dependent velocity field is to

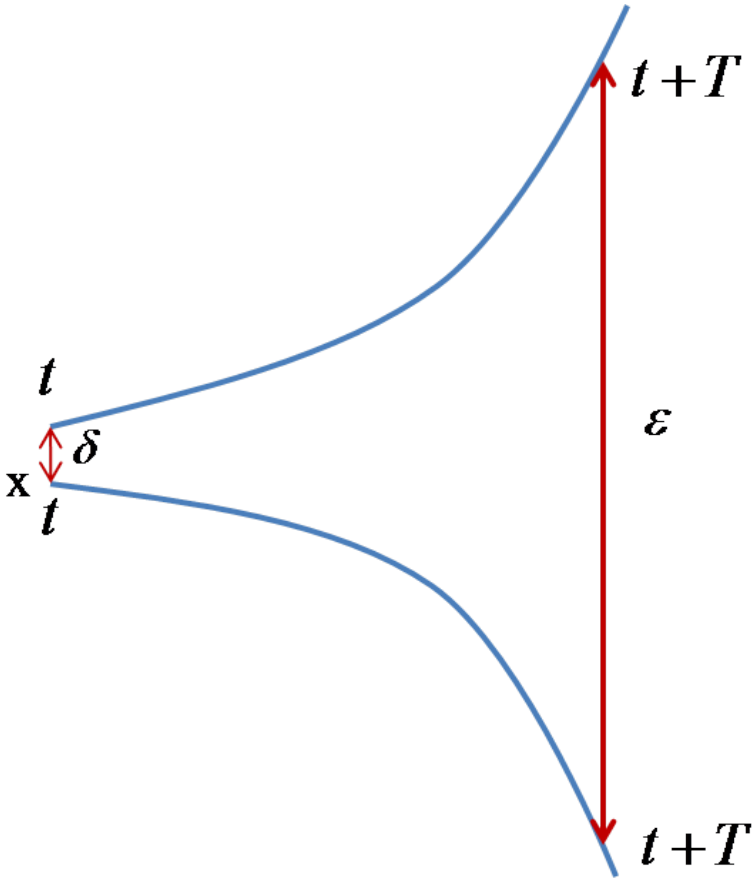


Figure 8.1: Separation of two points – Two points with initial distance δ evolves for time period T. The distance between two points after time T is ϵ.

measure the local stretching rate over a finite time interval [6, 109]. Fig. 8.1 shows the evolution of two nearby points with the initial distance δ, after time period T. The final distance between the two considered points is ϵ. The stretching rate between two points over the time interval T is computed as

$$FTLE(\mathrm{x}, t, T) = \frac{1}{|T|} ln \left(\frac{\epsilon}{\delta} \right). \tag{8.1}$$

When we compute the FTLEs numerically, however, we use a simpler approach which we explain next. In this case, for a given point $\mathbf{x} = \langle x(t), y(t) \rangle$, a flow map ϕ_T of $\mathbf{x}$ is obtained by evolving $\mathbf{x}$ over a time period $[t, t+T]$, according to the velocity components $\mathbf{v} = \langle u(x,y,t), v(x,y,t) \rangle$. Then the Jacobian matrix $J = \frac{d\phi_T(\mathbf{x})}{d\mathbf{x}}$ of the flow map ϕ_T is obtained and the finite time strain tensor of $\mathbf{v} = \langle u(x,y,t), v(x,y,t) \rangle$ along the trajectory $\mathbf{x} = \langle x(t), y(t) \rangle$ is obtained as

$$M = \frac{d\phi_T(\mathbf{x})^*}{d\mathbf{x}} \frac{d\phi_T(\mathbf{x})}{d\mathbf{x}}, \tag{8.2}$$

where A^* is the adjoint of A. Then the FTLE value at a point $\mathbf{x}$ over time T is given by

$$\sigma^T = \frac{1}{|T|} ln \sqrt{\lambda_{max}(M)}. \tag{8.3}$$

When the FTLEs are computed for the entire domain of the system, the ridges of the FTLE fields are extracted as the LCSs. The LCSs in the FTLE field are spatial gradient lines that are transverse to the direction of minimum curvature. These LCSs are suggested to act as pseudo barriers to mixing and transport of the system.

In another aspect, LCSs can be viewed as a stable or an unstable manifold of a hyperbolic fixed point. We will consider two points on either side of a stable manifold (red) of a hyperbolic fixed point as shown in Fig. 8.2 to explain this alternative view. Now, when we integrate the two points forward in time, the points move away from each other and they do not cross the stable manifold. The same scenario can be observed when we integrate two points on either side of an unstable manifold backward in time. In this case also, the points separate after a sufficient time, and they do not cross the unstable manifold. Now we will illustrate LCSs using numerical examples and the following two examples show the FTLE field for a closed form of system.

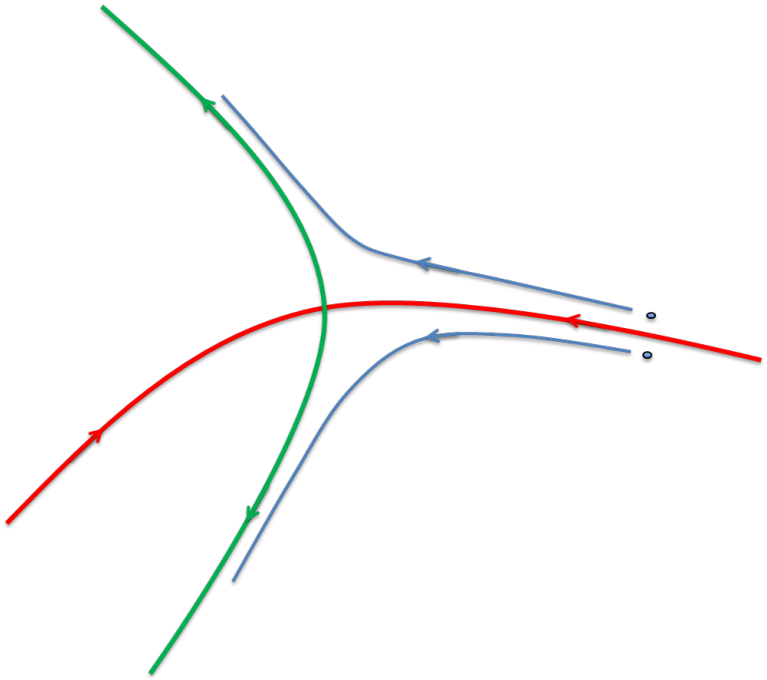


Figure 8.2: LCS as a stable manifold – Two points on either side of a stable manifold (red) in a hyperbolic fixed point are advected forward in time. After a sufficient time, they move away from each other and they do not cross the stable manifold.

8.1.1 Example: Double Gyre

For instance, if we consider the non-autonomous double gyre with the stream function

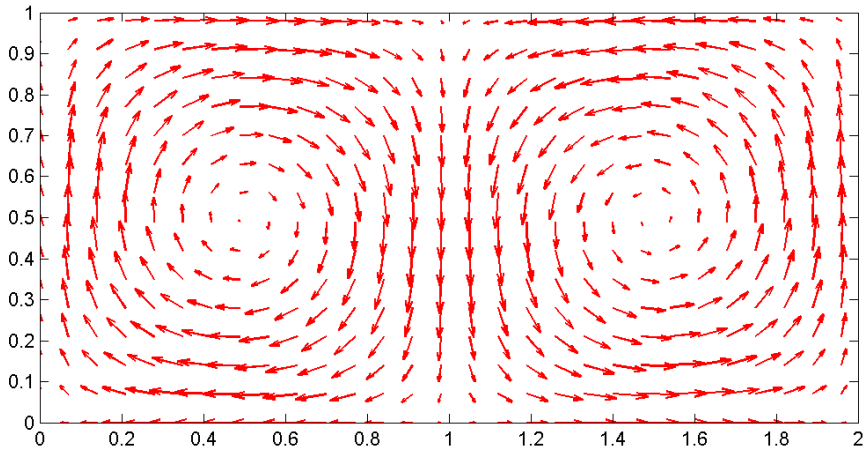
$$\psi(x, y, t) = C \sin(\pi f(x, t)) \sin(\pi y), \tag{8.4}$$

where $f(x, t) = \epsilon \sin(\omega t)x^2 + (1 - 2\epsilon \sin(\omega t))x$ over the domain $D = [0, 2] \times [0, 1]$. Then the corresponding vector field can be obtained as

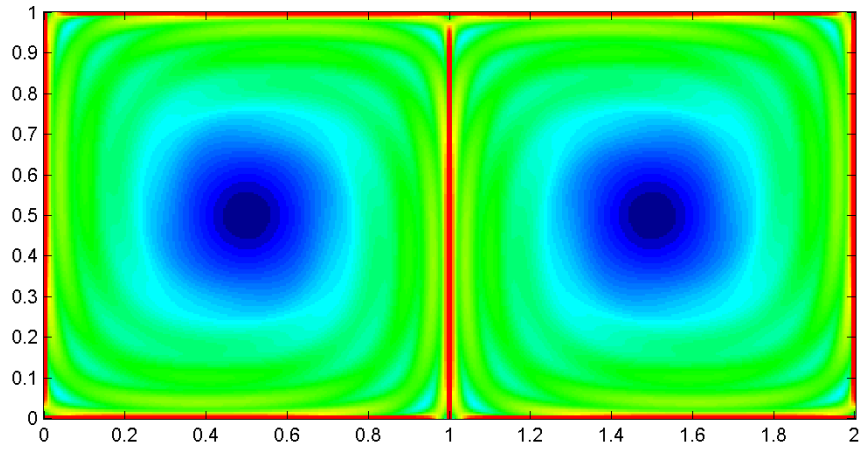
$$\langle u, v \rangle = \langle -\pi C \sin(\pi f(x, t)) \cos(\pi y), \pi C \cos(\pi f(x, t)) \sin(\pi y) \frac{\partial f}{\partial x} \rangle, \tag{8.5}$$

where C, ω and ϵ are parameters to be selected. If we choose $C = 1$ and $\epsilon = 0$, then the flow fields are time independent and the resulting system is known as an autonomous double gyre. We first compute the FTLE values over the domain $D = [0, 2] \times [0, 1]$

using the autonomous vector field as shown in image (a) in Fig. 8.3 with $T = 10$. That is, we use the same vector field for 10 time instances. Image (b) in Fig. 8.3 shows the FTLE field for the autonomous double gyre. Red represents high FTLE values whereas blue represents low FTLE values. The red ridges are the transport barriers which we often called LCSs.



(a)

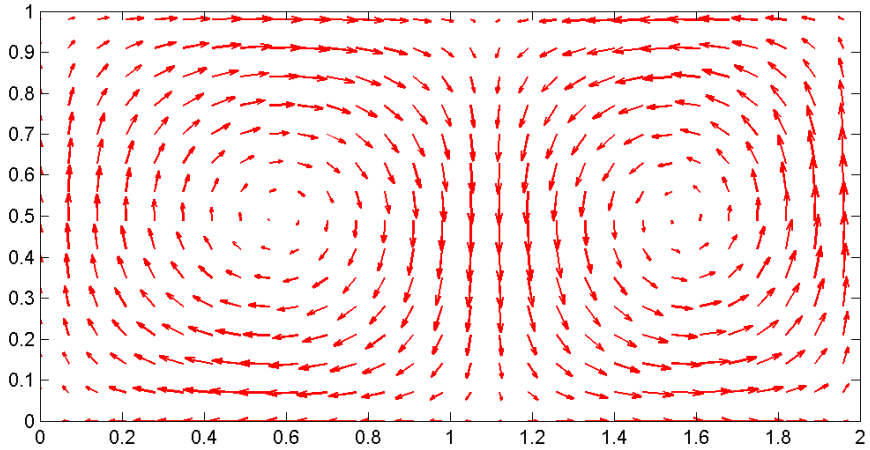


(b)

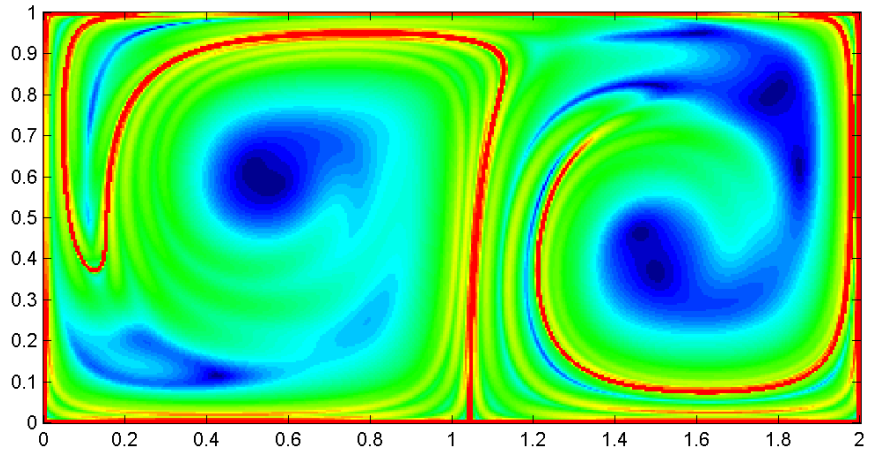
Figure 8.3: Autonomous double gyre – Image (a) shows the vector field of the autonomous double gyre and image (b) shows the computed FTLE field with $T = 10$. Red ridges are the transport barriers.

In [7], the authors have proven that flux across the LCS is close to zero and hence it is difficult for fluids in the system to cross the LCS. Therefore these LCSs act as

barriers to the mixing and transport. According to the above definition, the LCSs for the autonomous double gyre explain that the fluids in the domains $[0,1]\times[0,1]$ and $[1,2]\times[0,1]$ can not cross the barriers and hence they do not mix with each other.



(a)



(b)

Figure 8.4: Non-autonomous double gyre – The flow field for the non-autonomous double gyre at time $t = 13$ and the computed FTLE field for $T = 15$ are shown in images (a) and (b) respectively. Red ridges are the transport barriers.

We now change the parameters so that we have time dependent velocity fields. We choose the parameters $C = 0.1, \omega = \frac{2\pi}{10}$ and $\epsilon = 0.1$. The vector field that represents $\langle u, v\rangle$ at $t = 13$ is shown in image (a) of Fig. 8.4 and the computed FTLE field using

$\langle u, v \rangle$ with $T = 15$ is shown in the image (b) of Fig. 8.4. The red represents relatively high FTLE values while the blue represents relatively small FTLE values.

The autonomous and non-autonomous double gyres are examples of closed form of systems. However, in real applications, we may not have equations to determine velocity components and may have to use approximate velocity fields to compute FTLE values. Next we explain an application of FTLE on oceanic data which is not expressed in closed form of equations.

8.1.2 Example: Gulf of Mexico Oil Spill

In this example, we analyze the dynamics of the oil spill in the Gulf of Mexico which arose due to a failure of an oil well cap below the Deepwater Horizon rig on April 20, 2010. This was a man-made disaster. By July 15th, 2010 over 200 million gallons of crude oil spilled causing the deaths of thousands of animals. Some of the oil reached the Gulf Stream in just a few days was then carried around Florida and into the Atlantic Ocean.

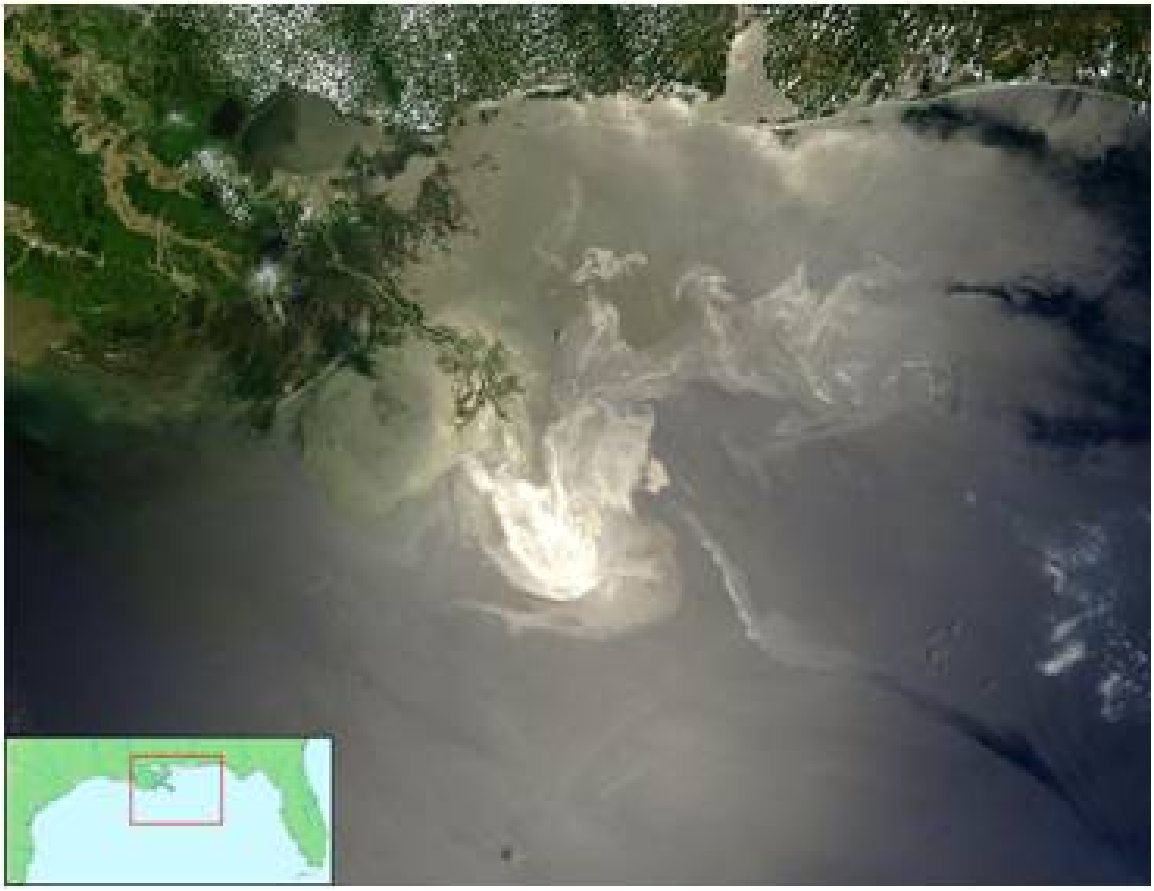


Figure 8.5: Satellite view – The satellite view of the Gulf of Mexico near Louisiana during the oil spill. The image was taken by NASA's Terra satellite on May 24, 2010. The spreading oil slick is visible and it can see in white.

In Fig. 8.5, a satellite image shows a view of the Gulf of Mexico off the coast of Louisiana. This image was taken by NASA's Terra satellite on May 24, 2010. Thirty-four days after the explosion, the oil had spread over a large area is clearly visible in white. Because the Gulf Stream carried oil into the Atlantic, it was predicted that a large portion of the oil would thus be spread into the Atlantic. However, the amount of oil spread in Atlantic was substantially reduced as there was a natural barrier to the oil flow due to a development of a natural eddy in the month of July 2010.

We now determine the Lagrangian coherent structures via computing FTLE to analyze the dynamics of the oil spill in the region of the Gulf of Mexico. To compute

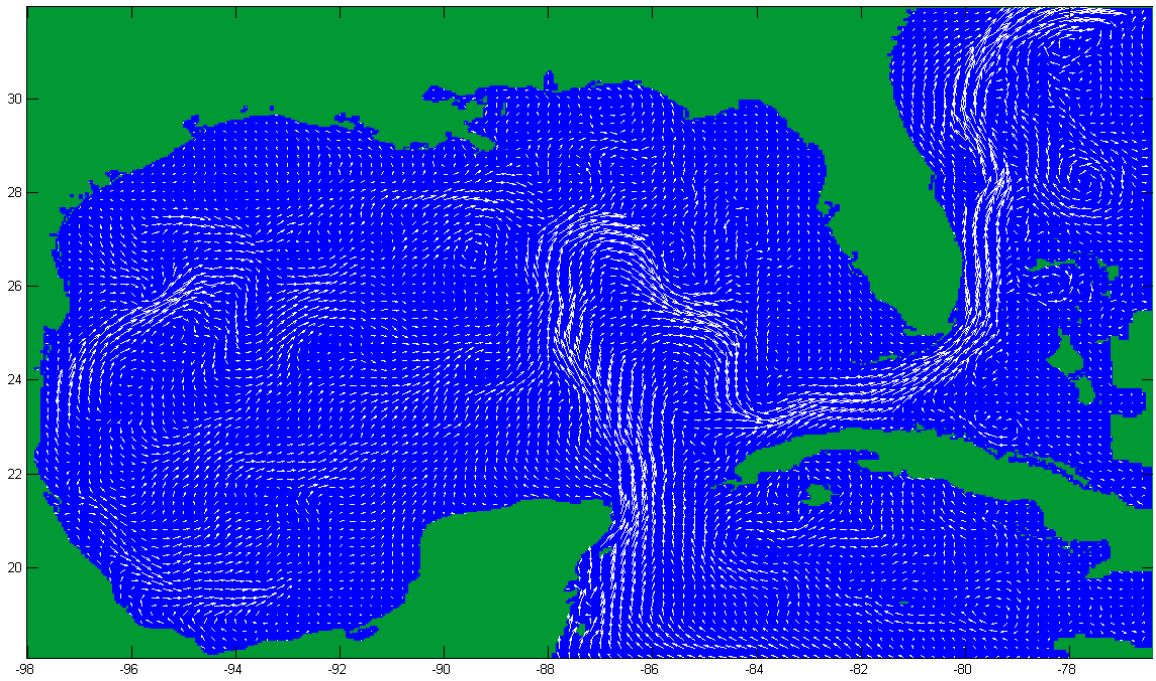


Figure 8.6: Gulf Velocity Field – Vector field from the HYCOM model is available in [116] on May 24, 2010. The image covers the velocity field in the area of the Gulf of Mexico where the longitudes and the latitudes are displaced. The Gulf Stream is clearly visible in the flow field and is close to the south of Louisiana near the source of the oil spill.

FTLE, the time dependent velocity fields of the considered region are required. Since the Gulf of Mexico is an interesting place for many researchers, a model to compute velocity fields was readily available. We obtained the velocity data of the Gulf of Mexico during the oil spill from the HYbrid Coordinate Ocean Model (HYCOM) which is freely available in [116]. The data set was generated, as explained in detail [117, 118], by developing a PDE model that describes the oceanic flow. The required data was collected from several institutes.

Fig. 8.6 shows the velocity field generated from the HYCOM model on May 24, 2010. The interested domain of the image covers the Gulf of Mexico region. In the velocity field, the Gulf Stream can be observed with relatively high velocity magnitudes very close to the south of Louisiana near the source of the oil spill.

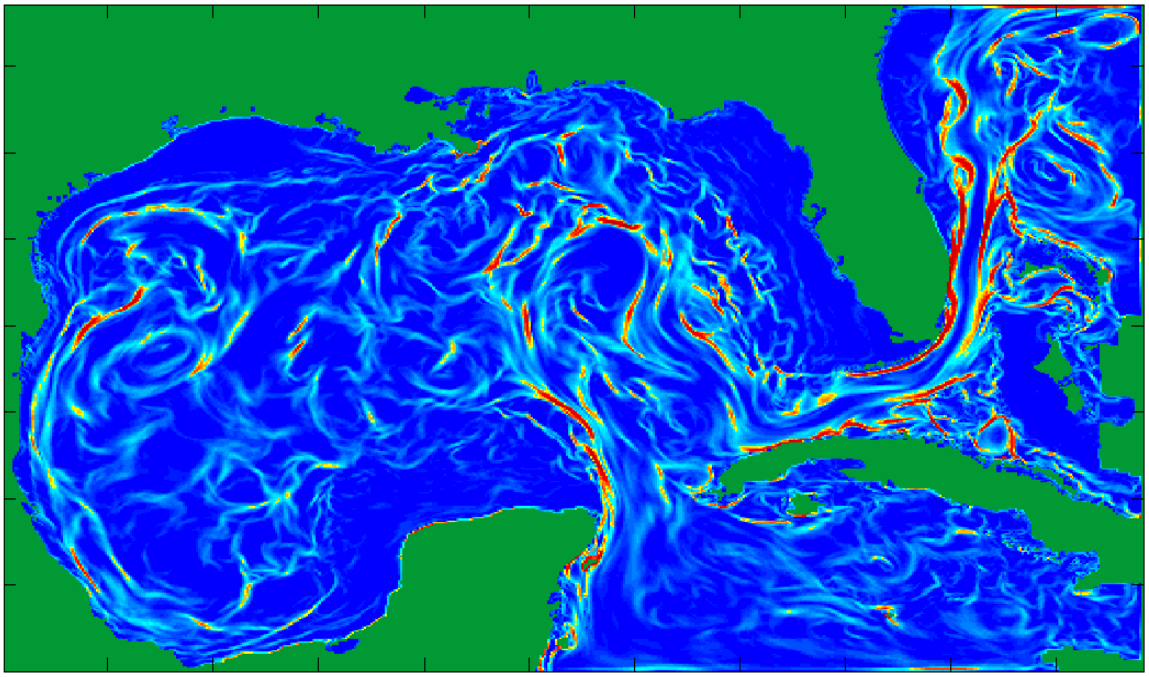


Figure 8.7: FTLE on May 24, 2010 – Computed FTLE field for the Gulf of Mexico on May 24, 2010 using the HYCOM data. The integration time is $T = 72$ hours and red ridges are the strong transport barriers for the fluid.

We now compute the FTLE values on the domain which is shown in Fig. 8.6. Unlike with double gyre, we do not have closed form of expressions for the velocity components u and v. Therefore, we use HYCOM data which comes as two arrays for the velocity components u and v sampled on a grid of the domain in each day. In other words, the velocity field is an array of $\mathbf{V}(\mathbf{x}, t)$, where $\mathbf{x}$ and t are the grid on the domain and the discrete time respectively. The time represents days; hence we interpolate data using cubic splines to get flow fields for every hour. Fig. 8.7 represents the FTLE for a late hour on May 24, 2010, using HYCOM velocity fields with $T = 72$ hours. The red represents relatively high FTLE values and the blue represents relatively low FTLE values. The land is represented by green. The oil on the same day is shown in Fig. 8.5 and the initial velocity field is shown in Fig. 8.6. The transport barriers appear in red. Flux through the barriers is zero or small. A detailed analysis of the FTLE field by evolving tracer particles can be found at [1].

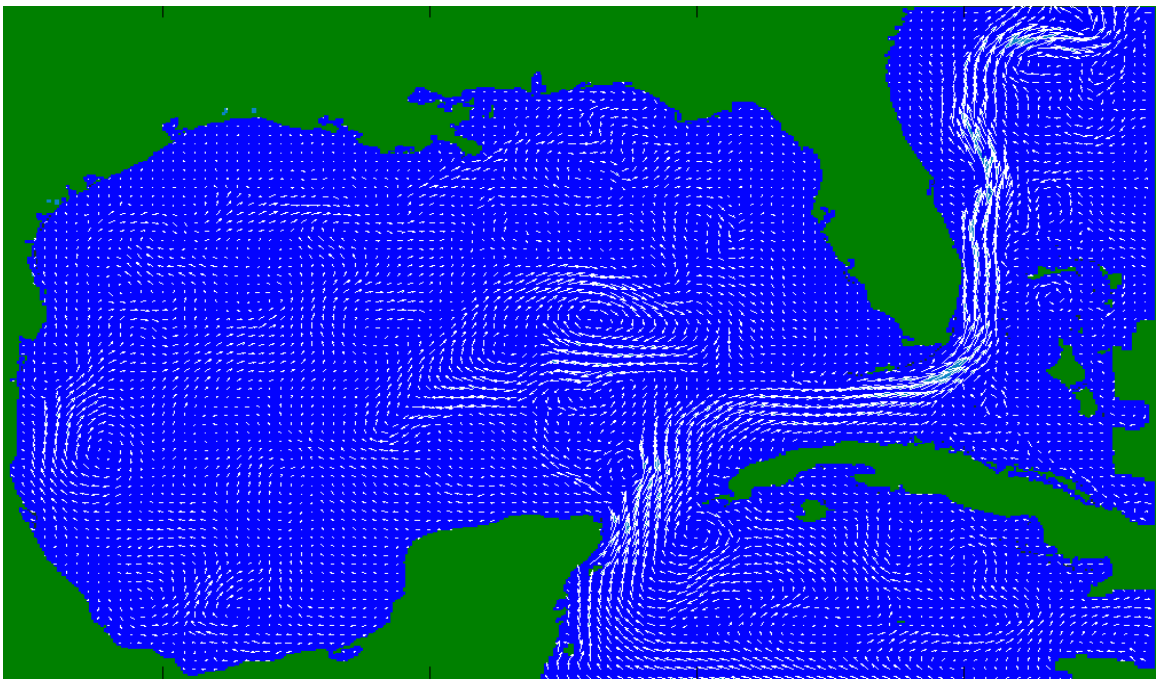


Figure 8.8: Vector field on July 27, 2010 – Image shows the HYCOM velocity field for July 27, 2010. There is a circulation in the center that is not connected to the Gulf Stream. Hence the central eddy does not transport oil to the Gulf Stream and that reduced the spread of oil.

As we mentioned, there were many initial predictions that the oil would move to Gulf Stream and then spread quickly. In [119], it was predicted the Gulf Stream would carry oil as far north as Cape Hatteras, North Carolina and the loop current that occurred in the center of Gulf would speed up this process. However, this phenomenon did not happen. Fig. 8.8 shows the HYCOM velocity field for July 27, 2010. It can be observed that there is a circulation loop in the center of the Gulf not connected to the Gulf stream. This circulation loop, known as the "Eddy Franklin," occurs predictably once a year. Because of the circulation of this eddy, oil did not move to the Gulf stream as expected earlier. However, in the flow field of May 24, 2010 in Fig. 8.6, the Gulf stream was connected to the flow in the center of Gulf and hence oil spread out quickly.

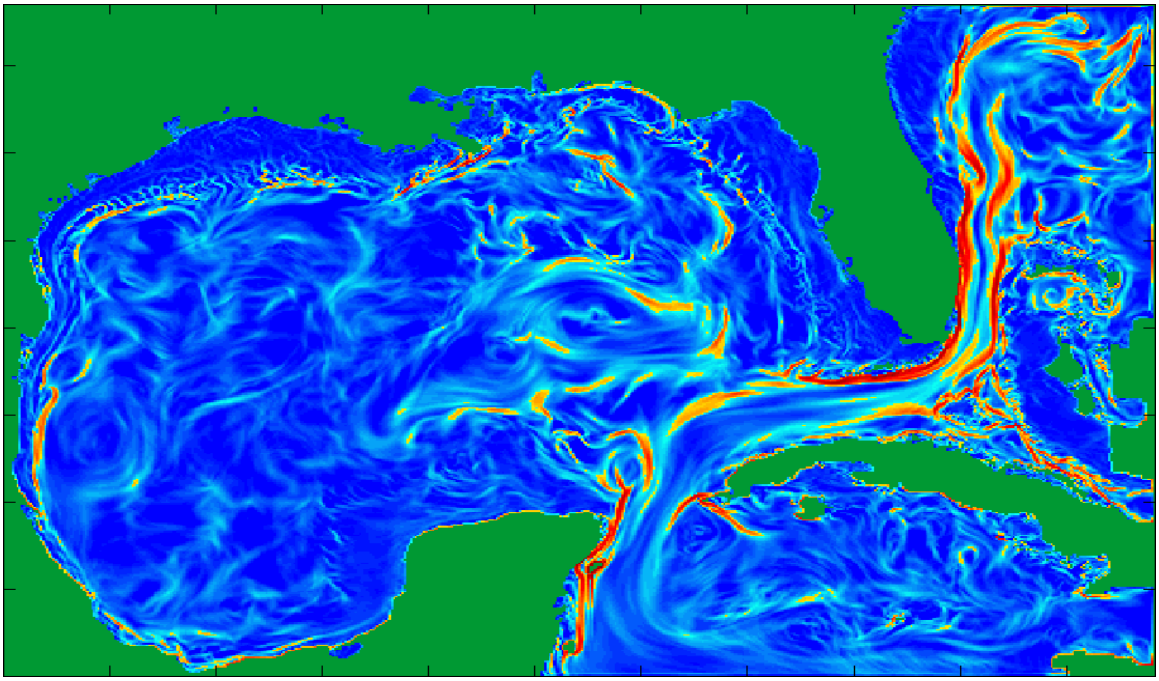


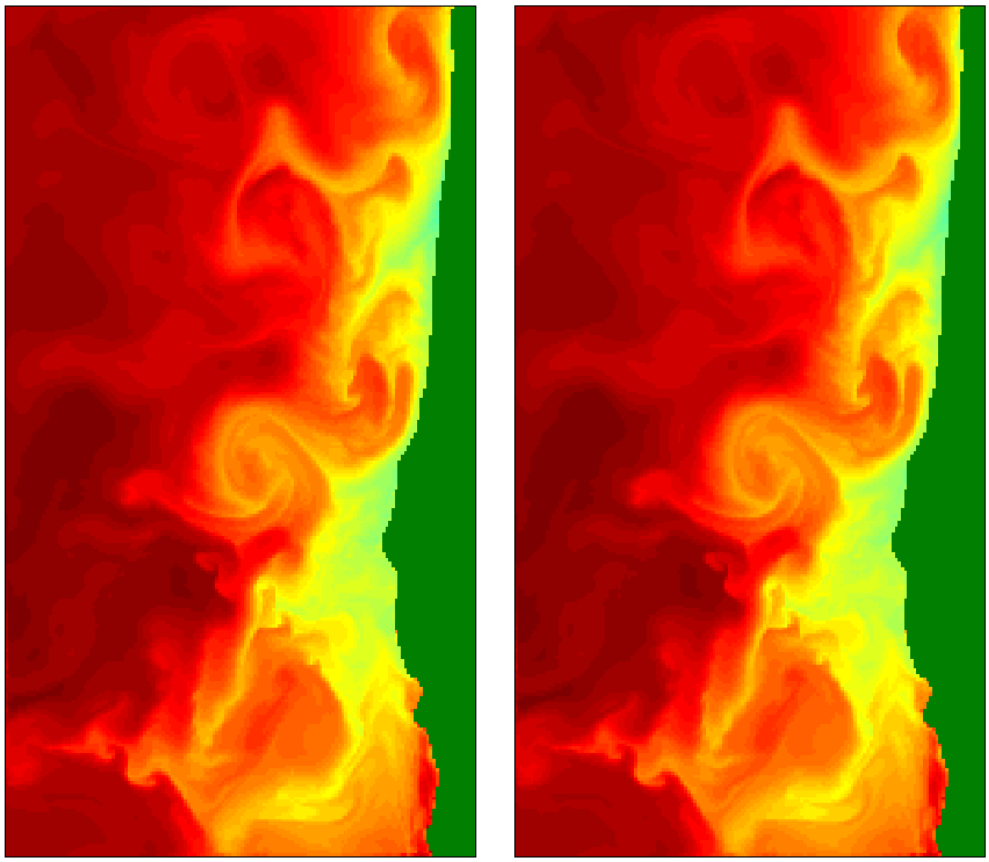
Figure 8.9: FTLE field on July 27, 2010 – As seen in the vector field in Fig. 8.8, the central eddy and the Gulf Stream are not connected thus reduce oil transport into the Gulf Stream. This can be observed in the FTLE field from the two orange ridges, in west of Florida which act as barriers to oil transport.

Again, we compute the FTLE field on the Gulf of Mexico domain using the velocity field in Fig. 8.8 for July 27, 2010 as the initial flow. Similar to the previous calculation, we set the evolution time at $T = 72$ hours. Fig. 8.9 shows the computed FTLE for the day of July 27, 2010. From the FTLE field, it is clearly visible that there is an orange ridge that acts as the transport barrier to the oil. This barrier reduced the amount of oil transported by the Gulf Stream to the Atlantic Ocean.

The above explanation and results reported in Sec. 8.1.1 give a strong validation for the FTLE approach of extracting LCSs of both a closed form system and a non-closed system. So far, in the computations of FTLEs we used readily available vector fields; however, in the rest of the work here, we are going to use computed vector fields from optical flow methods.

8.2 FTLE field for the SST data

For the SST data set which we introduced in Sec. 3.3.1, we do not have any closed form equations to compute the velocity fields. Therefore, we apply the optical flow methods to compute the velocity fields before we compute the FTLE values.

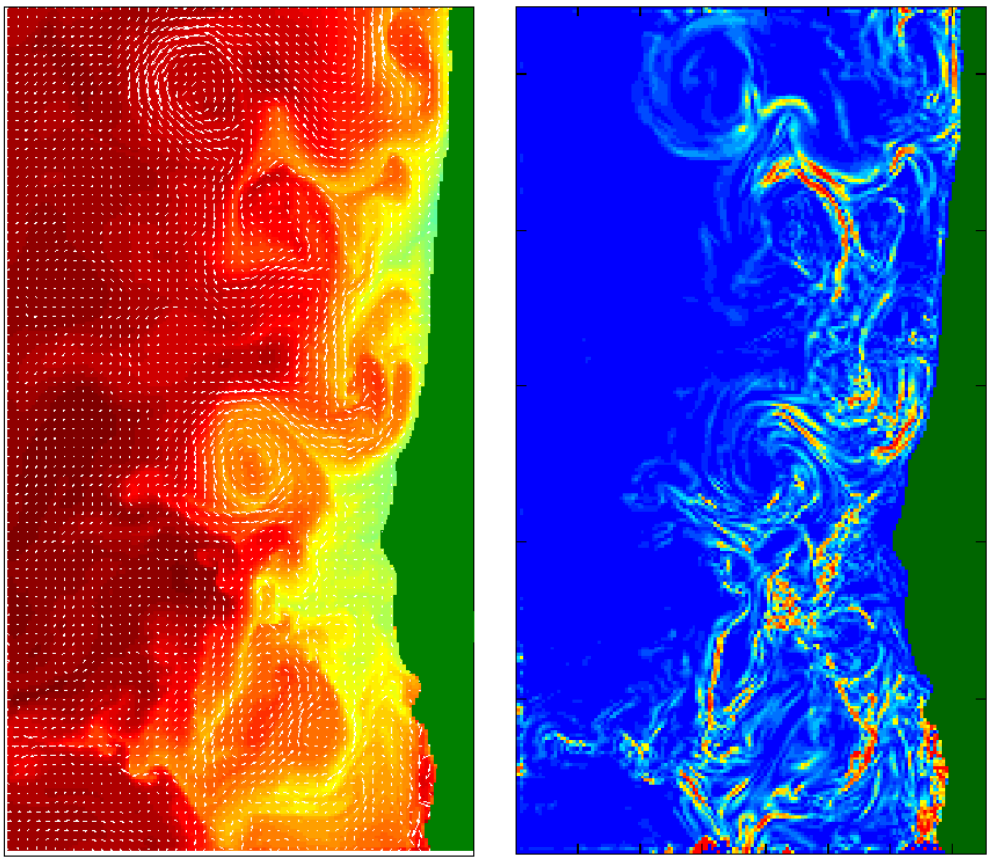


(a) image 1 (b) image 2

Figure 8.10: SST images – Images (a) and (b) show two consecutive images of the SST data set on August 1st, 2002. Red represents low temperature whereas orange represents high temperature. Green is the land.

Fig. 8.10 shows two time adjacent images of the SST time series on August 1st,

2002. First, we compute the FTLE field for the whole domain of the Sea Surface Temperature data from the computed velocity field using the conservation of image intensity data fidelity and the smoothness regularization term with the stream function formulation.

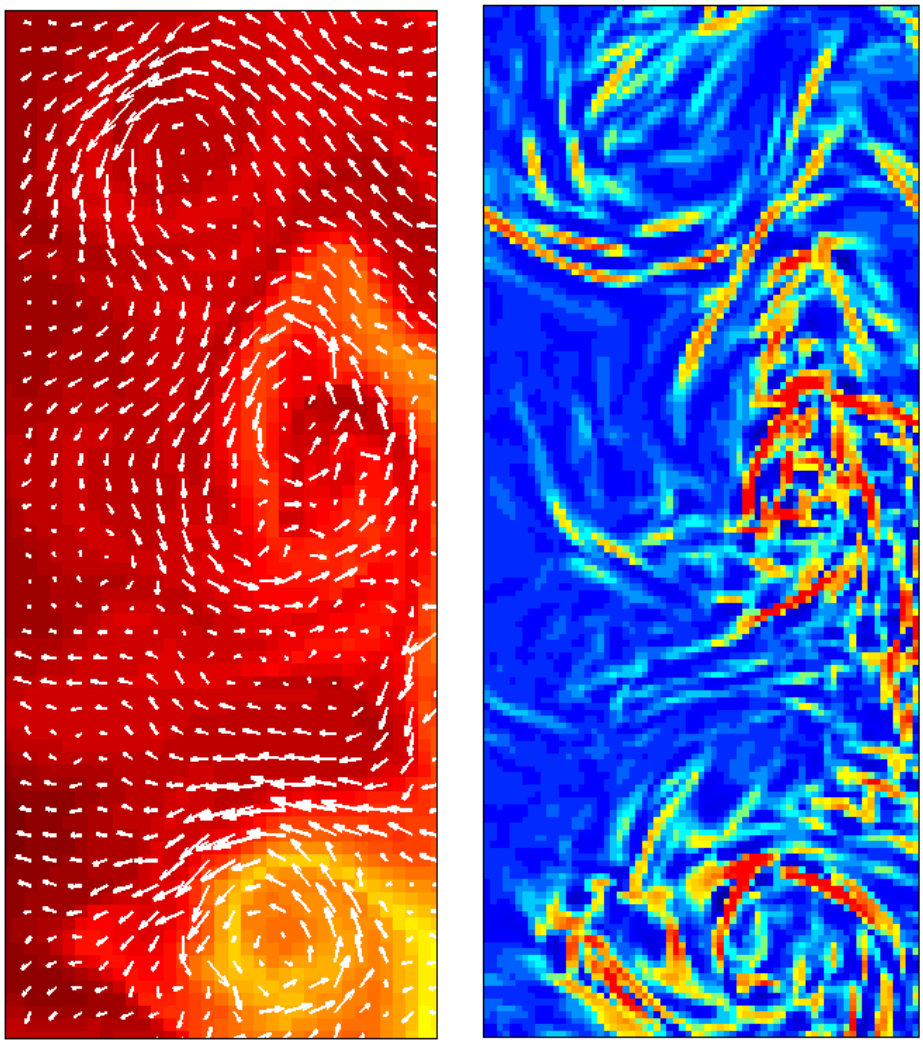


(a) (b)

Figure 8.11: SST flow and FTLE field – The computed flow field for the SST data on the image (a) in Fig. 8.10 from CI data fidelity and smoothness regularization term in stream function formulation is shown in image (a). The FTLE field with $T = 15$ on image (a) in Fig. 8.10 is shown in image(b).

When we compute the FTLE field, we use $T = 15$ hours and hence we have to reconstruct the velocity fields for the next 14 images. Therefore we apply the optical flow algorithm to reconstruct 15 velocity fields starting on the image (a) in Fig. 8.10. The computed FTLE field from the reconstructed velocity fields is shown in image

(a). Red represents the high FTLE value and blue represents the low FTLE values.



(a) (b)

Figure 8.12: SST flow and FTLE field – The computed flow field for the SST data and the FTLE field are shown in images (a) and (b) respectively

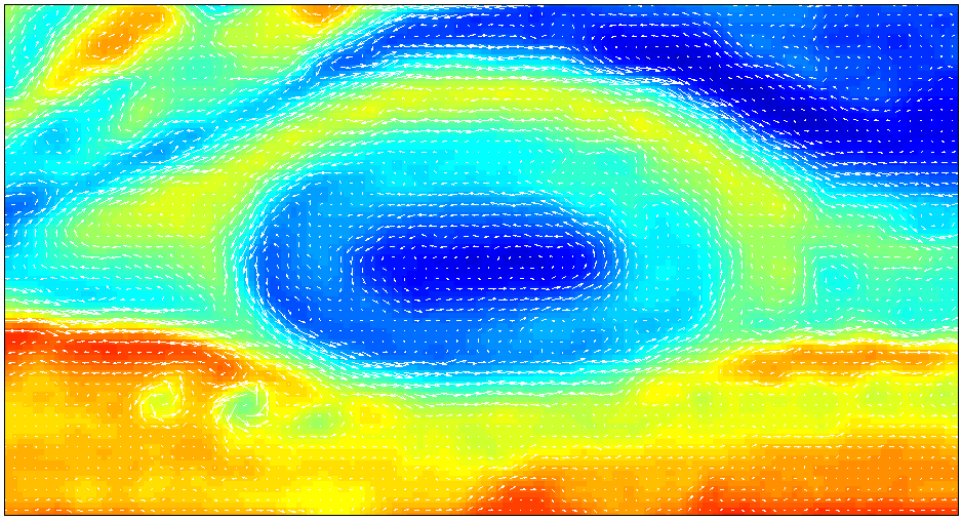
Next, we compute the FTLE field for the selected region in Fig. 3.8 to focus on the local structures in the SST data set. We use the computed flow from the multi-time step method with $n = 3$. In Fig. 8.12, the flow field obtained for the SST data on August 2nd, 2002 is shown in image (a) and the FTLE field obtained from those com-

puted vector fields is shown in (b). The blue represents relatively low FTLE values and the red represents relatively high FTLE values. The red ridges, LCSs, act as the mixing barriers to the heat.

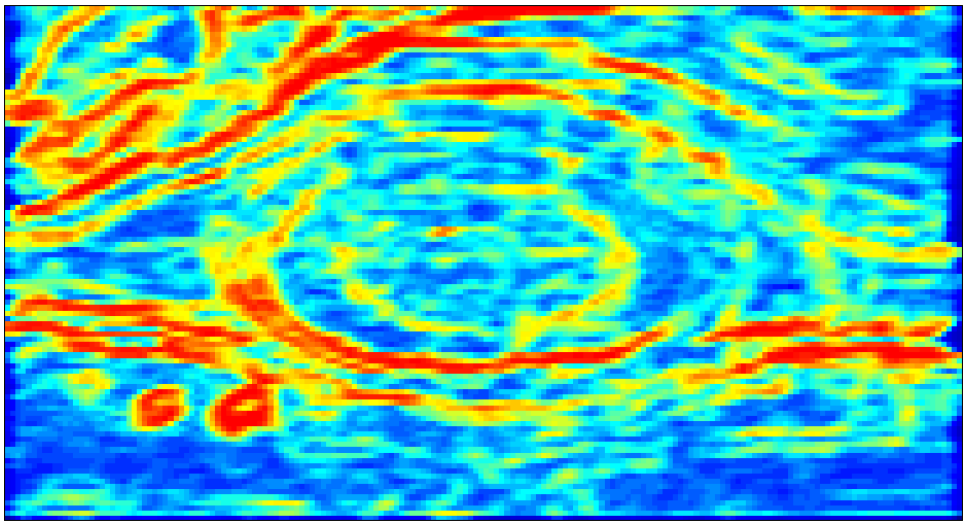
8.3 FTLE field for the Jupiter

Finally, we compute the FTLE field for the Jupiter data set which we introduced in Sec. 7.5. Three consecutive images are shown in Fig. 7.10. We use the quasi-geostrophic multi-time step method with step size 2. The computed flow on the first image is shown in image (a) of Fig. 8.13 and the computed FTLE field is shown in image (b) using the quasi-gestrophic flow fields. Here the integration time $T = 10$, which is equal to 2 Jovian days.

The LCSs appear in red which correspond to relatively high FTLE values. It can be observed that there is a ridge around the GRS and this verifies the barrier to the points inside the GRS.



(a)



(b)

Figure 8.13: GRS flow and FTLE field – The computed flow field for the GRS data and the FTLE field are shown in images (a) and (b) respectively. Flow field is computed from quasi-geostrophi multi-time step method with $n = 2$. The integration time for the FTLE is 2 Jovian days.

Chapter 9

Conclusion and Future work

In this thesis, we have adopted optical flow algorithms used for capturing rigid body motion to capture fluid motion in terms of stream functions providing many advantages. In the method of rigid body motion, the u-v method, the velocity components u and v are reconstructed by minimizing a functional that consists of a data fidelity and a regularization term. In our method, however, we have reconstructed a stream function and then computed the velocity components u and v from the simpletic gradient of the computed stream function. The stream function formulation has several advantages over the u-v formulation. The primary advantage of the stream function method is that numerically inexpensive as we need to solve the system for only one unknown function rather than two functions in u-v formulation. Also, the stream function formulation can be constructed for any existing energy functional in the u-v formulation. Furthermore, simple numerical schemes such as LU factorization and Gaussian elimination can be used to solve the Euler-Lagrange equation except for few regularization terms. The simple implementation is another advantage of the stream function method.

The stream function formulation plays an important role in regularization as well. In regularizing stream function formulation, we can impose regularity on the flow,

whereas in the u-v formulation we impose regularity in the flow components. As we have explained in [37], when the expected flow is sparse, we use L^1 regularization of flow components in the u-v formulation which enhances sparsity of u and v but not the sparsity of flow. In the stream function formulation, we use total variation of the stream function as the regularization term which leads to a sparse flow. Thus, in the stream function formulation, we can impose regularity on the governing flow while in the u-v formulation regularity is on the flow components.

Irrespective of the stream function formulation or the u-v formulation, when we include the total variation regularization term in the energy functional, the resulting Euler-Lagrange equations can not be solved as a linear system. Due to the existence of nonlinear terms in the Euler-Lagrange equations, we often use the gradient descent method to solve systems. As we have shown in Chapter 5, the convergence of the gradient descent approach is very slow. Therefore, we introduced the Lagged Diffusivity fixed point method to solve the nonlinear Euler-Lagrange equations. The introduction of the lagged diffusivity fixed point method to the optical flow computations has not only made the convergence faster but also enabled us to capture different flow patterns, which were not obtainable with the gradient descent method.

As we have seen in Chapters 3, 4 and 5, the development of the stream function formulation and the introduction of lagged diffusivity fixed point algorithm for the total variation regularized optical flow computations produced desired solutions. Our next goal was to apply these efficient algorithms to compute velocity fields in observed fluid systems. Although the results were promising, we noticed some discontinuities in the flow fields in the temporal direction. To overcome these temporal discontinuities, we developed the multi-time step method to compute n optical flow fields at a time from a sequence of images of an observed system. In this case we imposed the regularity in the time direction assuming that the characteristics of two consecutive stream functions are close to each other. Therefore, from this algorithm

we can emphasize the regularity not only spatially, but also in time. The creation of the multi-time step method served to avoid the discontinuities of the flow in time direction.

The satellite images obtained for large fluid systems such as in the ocean and the atmosphere are usually affected by the Coriolis force as the planet rotates. If the flow fields are reconstructed for images of this type, the resulting vector fields mush be corrected. Therein we have introduced quasi-static optical flow algorithm by incorporating quasi-static equations in the optical flow energy functional. In this way, we were able to eliminate the Coriolis effect from flow fields. This was shown in the results obtained from the synthetic data as well as real data discussed in Chapter 7.

Finally, we have computed the FTLE fields for the sea surface temperature data and Jupiter's atmospheric data to uncover the Lagrangian coherent structures from the reconstructed vector fields. The Lagrangian coherent structures for the Jupiter data set in Chapter 8 show promising transport barriers around the Great Red Spot, which validates the accuracy of the reconstructed velocity fields.

During the process of developing the above mentioned algorithms, the determination of the best regularization parameter α was always a critical step. According to the results discussed in Chapter 4, none of the common methods generally work on this regard. Therefore, more importantly, our future goal is to develop a method to determine best regularization parameter for a general optical flow algorithm. The method anticipated may focus on not only a single parameter but also spatially dependent parameters.

In addition to the selection of regularization parameter, we have experienced another difficulty with determining the optimal step size n in the multi-time step method. The lower n causes discontinuity in time and the larger n causes discontinuities between the n^{th} th flow field and $(n+1)^{th}$ flow field. Therefore, the determination

of optimal n would be a necessary task in future work.

Furthermore, we expect to expand our optical flow algorithms to compute optical flow fields from images with large displacements. Sometimes in real applications, two consecutive images are taken with a large time interval or the magnitudes of the velocity fields are high, and hence the displacement is large. Currently, we interpolate the images by adding artificial frames between two actual frames. It would be great if we could perform both steps at once from minimizing the energy functional. Some work about large displacement optical flow methods can be found in [120, 121].

In the quasi-static optical flow method, we used the gradient descent algorithm as the numerical scheme. There were some instances where the algorithm did not converge even after 100,000 iterations as discussed in Chapter 7. Perhaps, incorporating a faster numerical scheme in the quasi-static optical flow method would be a task of importance in the future.

Further, we can consider the changes of level sets [122] of image brightness between two consecutive images rather than considering the pixel wise brightness changes. One advantage of this approach is to compute the flow field of a patch with same color where the usual optical flow methods have difficulty computing such flows with only two images. Fig. 9.1 is the contour plot of the image (a) in Fig. 6.9 which represents Jupiter's atmospheric motion.

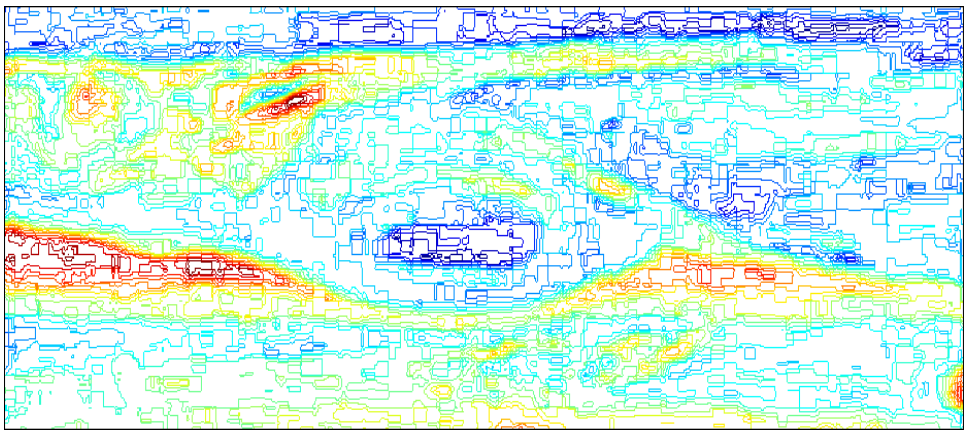


Figure 9.1: Contour plot of GRS – Image shows a contour plot of the Jupiter's atmospheric motion which is shown in image (a) of Fig. 6.9.

Bibliography

[1] E.M. Bollt, A. Luttman, S. Kramer, and R. Basnayake. Measurable dynamics analysis of transport in the Gulf of Mexico during the oil spill. *International Journal of Bifurcation and Chaos*, 22(03), 2012.

[2] FJ Beron-Vera, MJ Olascoaga, and GJ Goni. Oceanic mesoscale eddies as revealed by lagrangian coherent structures. *Geophysical Research Letters*, 35(12), 2008.

[3] FJ Beron-Vera, Michael G Brown, MJ Olascoaga, Irina I Rypina, H Koçak, and Ilya A Udovydchenkov. Zonal jets as transport barriers in planetary atmospheres. *arXiv preprint arXiv:0803.2893*, 2008.

[4] Sushil Shetty and Philip S Marcus. Changes in jupiter's great red spot (1979–2006) and oval ba (2000–2006). *Icarus*, 210(1):182–201, 2010.

[5] Francisco J Beron-Vera, María J Olascoaga, Michael G Brown, and Huseyin Koçak. Zonal jets as meridional transport barriers in the subtropical and polar lower stratosphere. *Journal of the Atmospheric Sciences*, 69(2):753–767, 2012.

[6] G. Haller. Lagrangian coherent structures from approximate velocity data. *Physics of fluids*, 14:1851, 2002.

[7] Shawn C Shadden, Francois Lekien, and Jerrold E Marsden. Definition and properties of lagrangian coherent structures from finite-time lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, 212(3):271–304, 2005.

[8] Gary Froyland, Naratip Santitissadeekorn, and Adam Monahan. Transport in time-dependent dynamical systems: Finite-time coherent sets. *arXiv preprint arXiv:1008.1613*, 2010.

[9] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[10] University of otago, computer vision homepage, available at. `http://www.cs.otago.ac.nz/research/vision/`. [Online; accessed 25-Oct-2013].

[11] Per Christian Hansen, James G Nagy, and Dianne P O'leary. *Deblurring images: matrices, spectra, and filtering*, volume 3. Siam, 2006.

[12] Per Christian Hansen. *Discrete inverse problems: insight and algorithms*, volume 7. SIAM, 2010.

[13] P. C. Hansen. The L-curve and its use in the numerical treatment of inverse problems. In P. Johnston, editor, *Computational Inverse Problems in Electrocardiology, Advances in Computational Bioengineering*, pages 119–142. WIT Press, 2000.

[14] D. Krawczyk-Stańdo and M. Rudnicki. The use of L-curve and U-curve in inverse electromagnetic modelling. *Intelligent Computer Techniques in Applied Electromagnetics*, pages 73–82, 2008.

[15] D. Krawczyk-StańDo and M. Rudnicki. Regularization parameter selection in discrete ill-posed problems—the use of the U-curve. *International Journal of Applied Mathematics and Computer Science*, 17(2):157–164, 2007.

[16] I.M. Gelfand and S.V. Fomin. *Calculus of variations*. Dover publications, 2000.

[17] Bernard Dacorogna, Bernard Dacorogna, Bernard Dacorogna, Bernard Dacorogna, Egypt Mathematician, and Great Britain. *Introduction to the Calculus of Variations*. World Scientific, 2004.

[18] Jim L Mitchell, Reta F Beebe, Andrew P Ingersoll, and Glenn W Garneau. Flow fields within jupiter's great red spot and white oval bc. *Journal of Geophysical Research: Space Physics (1978–2012)*, 86(A10):8751–8757, 1981.

[19] Xylar S Asay-Davis, Philip S Marcus, Michael H Wong, and Imke de Pater. Jupiter's shrinking great red spot and steady oval ba: Velocity measurements with the 'advection corrected correlation image velocimetry'automated cloud-tracking method. *Icarus*, 203(1):164–188, 2009.

[20] G. M. Quénot, J. Pakleza, and T. A. Kowalewski. Particle image velocimetry with optical flow. *Experiments in Fluids*, 25:177–189, 1998.

[21] Georges M Quénot, Jaroslaw Pakleza, and Tomasz A Kowalewski. Particle image velocimetry using optical flow for image analysis. In *8th Int. Symposium on Flow Visualization*, pages 47–1, 1998.

[22] AM Fincham and GR Spedding. Low cost, high resolution dpiv for measurement of turbulent fluid flow. *Experiments in Fluids*, 23(6):449–462, 1997.

[23] D. Auroux. Extraction of velocity fields for geophysical fluids from a sequence of images. In *Acoustics, Speech, and Signal Processing, IEEE Proceedings on (ICASSP)*, pages 961–964, 2009.

[24] D. Auroux and J. Fehrenbach. Identification of velocity fields for geophysical fluids from a sequence of images. *Experiments in Fluids*, 50(2):313–328, 2010.

[25] W. Bresky and J. Daniels. The feasibility of an optical flow algorithm for estimating atmospheric motion. In *Proceedings of the Eighth Int. Winds Workshop, Beijing, China*, pages 24–28, 2006.

[26] C. Cassisa, V. Prinet, L. Shao, S. Simoens, and C. L. Liu. Optical flow robust estimation in a hybrid multi-resolution MRF framework. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, pages 793–796, 2008.

[27] Aaron Luttman, Erik Bollt, and Jason Holloway. An optical flow approach to analyzing species density dynamics and transport. *J. Comput. Math.*, 30(3):249–261, 2012.

[28] Aaron Luttman, Erik Bollt, Ranil Basnayake, and Sean Kramer. A stream function approach to optical flow with applications to fluid transport dynamics. In *Proc. Appl. Math. Mechanics*, volume 11, pages 855–856, 2011.

[29] T. Corpetti, É. Mémin, and P. Pérez. Adaptation of standard optic flow methods to fluid motion. In *9th Int. Symp. Flow Visualisation*, pages 1–10, 2000.

[30] T. Corpetti, É. Mémin, and P. Pérez. Estimating fluid optical flow. In *ICPR*, pages 7045–7048, 2000.

[31] T. Corpetti, É. Mémin, and P. Pérez. Dense estimation of fluid flows. *IEEE Trans. Pattern Anal. Machine Intelligence*, 24(3):365–380, 2002.

[32] T. Corpetti, É. Mémin, and P. Pérez. Dense motion analysis in fluid imagery. In A. Heyden, editor, *Proc. 7th Eur. Conf. Computer Vision*, pages 676–691, 2002.

[33] É. Mémin and T. Corpetti. Dense fluid flow estimation. Technical report, INRIA, 2000.

[34] D. Suter. Motion estimation and vector splines. In *Proc. Conf. Comp. Vision Pattern Rec*, pages 939–942. IEEE, 1994.

[35] T. Kohlberger, E. Mémin, and C. Schnorr. Variational dense motion estimation using the helmholtz decomposition. In L. D. Griffin and M. Lillholm, editors, *Scale Space '03*, volume 2695, pages 432–448, Isle of Skye, UK, 2003.

[36] Aaron Luttman, Erik Bollt, Ranil Basnayake, and Sean Kramer. A stream function approach to optical flow with applications to fluid transport dynamics. *PAMM*, 11(1):855–856, 2011.

[37] Aaron Luttman, Erik Bollt, Ranil Basnayake, and Sean Kramer. A framework for estimating potential fluid flow from digital imagery. *Chaos*, 23(3), 2013.

[38] Timo Kohlberger, Étienne Mémin, and Christoph Schnörr. Variational dense motion estimation using the helmholtz decomposition. In *Scale Space Methods in Computer Vision*, pages 432–448. Springer, 2003.

[39] Curtis R Vogel. *Computational methods for inverse problems*, volume 10. Siam, 2002.

[40] Christoph Brune, Helmut Maurer, and Marcus Wagner. Detection of intensity and motion edges within optical flow via multidimensional control. *SIAM Journal on Imaging Sciences*, 2(4):1190–1210, 2009.

[41] J. Weickert, A. Bruhn, N. Papenberg, and T. Brox. Variational optic flow computation: From continuous models to algorithms. In L. Alvarez, editor, *International Workshop on Computer Vision and Image Analysis, IWCVIAí03, Las Palmas de Gran Canaria*, 2003.

[42] Ashish Doshi and Adrian G Bors. Navier-stokes formulation for modelling turbulent optical flow. In *BMVC*, pages 1–10, 2007.

[43] Hurricanes/tropical cyclones, nasa, available at. `http://www.nasa.gov/mission_pages/hurricanes/main/#.UnO-TOL-VEll`. [Online; accessed 31-Oct-2013].

[44] Solar system exploration, nasa, available at. `http://solarsystem.nasa.gov/multimedia/display.cfm?Category=Planets&IM_ID=13347l`. [Online; accessed 31-Oct-2013].

[45] National snow and ice data center, available at. `https://nsidc.org/cryosphere/arctic-meteorology/factors_affecting_climate_weather.html`. [Online; accessed 31-Oct-2013].

[46] Coriolis effect, available at. `http://education.nationalgeographic.com/education/encyclopedia/coriolis-effect/?ar_a=1`. [Online; accessed 11-01-2013].

[47] Benoit Cushman-Roisin and Jean-Marie Beckers. *Introduction to geophysical fluid dynamics: physical and numerical aspects*, volume 101. Access Online via Elsevier, 2011.

[48] Department of water resorces, available at. `http://publicaffairs.water.ca.gov/swp/swptoday.cfm`. [Online; accessed 12-Sep-2012].

[49] C. R. Vogel. *Computational Methods for Inverse Problems*. Frontiers in Mathematics. SIAM, 2002.

[50] James R Cooper and Nicola Ritter. Optical flow for validating medical image registration. In *SIP*, pages 502–506, 2003.

[51] Amir A Amini. A scalar function formulation for optical flow. In *Computer Vision—ECCV'94*, pages 123–131. Springer, 1994.

[52] Caren Marzban and Scott Sandgathe. Optical flow for verification. *Weather & Forecasting*, 25(5), 2010.

[53] S Das Peddada and Robert McDevitt. Least average residual algorithm (lara) for tracking the motion of arctic sea ice. *Geoscience and Remote Sensing, IEEE Transactions on*, 34(4):915–926, 1996.

[54] Klaus Janschek, V Tchernykh, and M Beck. Performance analysis for visual planetary landing navigation using optical flow and dem matching. In *Proceedings of the AIAA Guidance, Navigation and Control Conference*, pages 21–24, 2006.

[55] Ranil Basnayake, Aaron Luttman, and Erik Bollt. A lagged diffusivity method for computing total variation regularized fluid flow. *Contemporary Mathematics*, 586:59–66, 2013.

[56] J. J. Osborne, A. L. Kurapov, G. D. Egbert, and P. M. Kosro. Spatial and temporal variability of the m2 internal tide generation and propagation on the oregon shelf. *Journal of Physical Oceanography*, 41(11):2037–2062, 2013/05/29 2011.

[57] J. J. Osborne, A. L. Kurapov, G. D. Egbert, and P. M. Kosro. Spatial and temporal variability of the M2 internal tide generation and propagation on the Oregon shelf. *Journal of Physical Oceanography*, in press (DOI: 10.1175/JPO-D-11-02.1), 2011.

[58] E. M. Bollt, A. Luttman, S. Kramer, and R. Basnayake. Measurable dynamics analysis of transport in the gulf of mexico during the oil spill. *Int. J. Bifurc. Chaos*, 22(3):1–12, 2012.

[59] Han-Dol Kim et al. Coms, the new eyes in the sky for geostationary remote sensing. *REMOTE SENSING–ADVANCED TECHNIQUES AND PLATFORMS*, page 235, 2012.

[60] M. Schmidt, G. Fung, and R. Rosales. Fast optimization methods for l1 regularization: A comparative study and two new approaches. *Machine Learning: ECML 2007*, pages 286–297, 2007.

[61] A Luttman. *Introduction to Inverse Problems*. Course Notes, Clarkson University, 2010.

[62] Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[63] Andreĭ Tikhonov. *Numerical methods for the solution of ill-posed problems*.

[64] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In T. Pajdla and J. Matas, editors, *Proc. 8th Eur. Conf. on Computer Vision*, volume 4, pages 25–36, 2004.

[65] Sergey I Kabanikhin. *Inverse and Ill-Posed Problems: Theory and Applications*, volume 55. De Gruyter, 2011.

[66] D. Koppel, C.-M. Tsai, and Y.-F. Wang. Regularizing optical-flow computation using tensor theory and complex analysis. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08. IEEE Computer Society Conference on*, pages 1–6, 2008.

[67] John L Barron, David J Fleet, and Steven S Beauchemin. Performance of optical flow techniques. *International journal of computer vision*, 12(1):43–77, 1994.

[68] Brendan McCane, Kevin Novins, D Crannitch, and Ben Galvin. On benchmarking optical flow. *Computer Vision and Image Understanding*, 84(1):126–143, 2001.

[69] P. C. Hansen. Analysis of discrete ill-posed problems by means of the L-curve. *SIAM Review*, 34:561–580, 1992.

[70] Per Christian Hansen and Dianne Prost O'Leary. The use of the l-curve in the regularization of discrete ill-posed problems. *SIAM Journal on Scientific Computing*, 14(6):1487–1503, 1993.

[71] D. Krawczyk-Stado and M. Rudnicki. Regularization parameter selection in discrete ill-posed problems – the use of the U-curve. *Int. J. Appl. Math. Comput. Sci.*, 17:157–164, 2007.

[72] D. Krawczyk-Stado and M. Rudnicki. The use of L-curve and U-curve in inverse electromagnetic modelling. *Intell. Comput. Tech. Appl. Electromagn.*, 119:73–82, 2008.

[73] G. Golub, M. Heath, and G. Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21:215–223, 1979.

[74] M. A. Lukas. Strong robust generalized cross-validation for choosing the regularization parameter. *Inverse Problems*, 24:034006, 2008.

[75] Daniela Calvetti, Per Christian Hansen, and Lothar Reichel. L-curve curvature bounds via lanczos bidiagonalization. *Electronic Transactions on Numerical Analysis*, 14:20–35, 2002.

[76] C. R. Vogel. Non-convergence of the L-curve regularization parameter selection method. *Inverse Problems*, 12:535–548, 1996.

[77] M. Hanke. Limitations of the L-curve method in ill-posed problems. *BIT*, 36:287–301, 1996.

[78] Per Christian Hansen. *The L-curve and its use in the numerical treatment of inverse problems*. IMM, Department of Mathematical Modelling, Technical Universityof Denmark, 1999.

[79] Daniela Calvetti, Lothar Reichel, and A Shuibi. L-curve and curvature bounds for tikhonov regularization. *Numerical Algorithms*, 35(2-4):301–314, 2004.

[80] David Suter. Motion estimation and vector splines. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pages 939–942. IEEE, 1994.

[81] Aaron Luttman, Erik Bollt, Ranil Basnayake, and Sean Kramer. A stream function approach to optical flow with applications to fluid transport dynamics. *PAMM*, 11(1):855–856, 2011.

[82] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total-variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60:259–268, 1992.

[83] L. Rudin and S. Osher. Total variation based image restoration with free local constraints. In *Proc. 1st IEEE ICIP*, volume 1, pages 31–35, 1994.

[84] Yunhai Xiao and Junfeng Yang. A fast algorithm for total variation image reconstruction from random projections. *arXiv preprint arXiv:1001.1774*, 2010.

[85] Junzhou Huang, Shaoting Zhang, and Dimitris Metaxas. Efficient mr image reconstruction for compressed mr imaging. *Medical Image Analysis*, 15(5):670–679, 2011.

[86] Marius Drulea and Sergiu Nedevschi. Total variation regularization of local-global optical flow. In *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pages 318–323. IEEE, 2011.

[87] Andreas Wedel, Thomas Pock, Christopher Zach, Horst Bischof, and Daniel Cremers. An improved algorithm for tv-l 1 optical flow. In *Statistical and Geometrical Approaches to Visual Motion Analysis*, pages 23–45. Springer, 2009.

[88] Curtis R Vogel and Mary E Oman. Iterative methods for total variation denoising. *SIAM Journal on Scientific Computing*, 17(1):227–238, 1996.

[89] Donald Geman and Chengda Yang. Nonlinear image recovery with half-quadratic regularization. *Image Processing, IEEE Transactions on*, 4(7):932–946, 1995.

[90] Tony F Chan and Pep Mulet. On the convergence of the lagged diffusivity fixed point method in total variation image restoration. *SIAM journal on numerical analysis*, 36(2):354–367, 1999.

[91] Dominique Heitz, Etienne Mémin, and Christoph Schnörr. Variational fluid flow measurements from image sequences: synopsis and perspectives. *Experiments in fluids*, 48(3):369–393, 2010.

[92] Tianshu Liu and Lixin Shen. Fluid flow and optical flow. *Journal of Fluid Mechanics*, 614:253–291, 2008.

[93] Ashish Doshi and Adrian G Bors. Robust processing of optical flow of fluids. *Image Processing, IEEE Transactions on*, 19(9):2332–2344, 2010.

[94] T. Corpetti, D. Heitz, G. Arroyo, É. Mémin, and A. Santa-Cruz. Fluid experimental flow estimation based on an optical-flow scheme. *Experiments in Fluids*, 40(1):80–97, 2006.

[95] Ranil Basnayake and Erik M Bollt. A multi-time step method to compute optical flow with scientific priors for observations of a fluidic system. *Ergodic Theory, Open Dynamics, and Coherent Structures*, 84(4):59–79, 2014.

[96] Office of sattellite operation, 2011.

[97] Jupiter's cloud movements, available at. `https://www.youtube.com/watch?v=C_1hihXOAjw`. [Online; accessed 09-01-2013].

[98] Coriolis effect , available at. `http://en.wikipedia.org/wiki/Coriolis_effect`. [Online; accessed 09-01-2013].

[99] Solar system exploration, nasa , available at. `http://solarsystem.nasa.gov/planets/profile.cfm?Object=Jupiter`. [Online; accessed 11-01-2013].

[100] Jupiter: Largest planet of the solar system , available at. `http://www.space.com/7-jupiter-largest-planet-solar-system.html`. [Online; accessed 11-01-2013].

[101] Philip S Marcus. Numerical simulation of jupiter's great red spot. *Nature*, 331(6158):693–696, 1988.

[102] Photojournal , available at. `http://photojournal.jpl.nasa.gov/catalog/PIA02829`. [Online; accessed 11-01-2013].

[103] Gary Froyland and Kathrin Padberg. Almost-invariant sets and invariant manifolds—connecting probabilistic and geometric descriptions of coherent structures in flows. *Physica D: Nonlinear Phenomena*, 238(16):1507–1523, 2009.

[104] Erik M Bollt and Naratip Santitissadeekorn. *Applied and Computational Measurable Dynamics*, volume 18. SIAM, 2013.

[105] Tian Ma and Erik Bollt. Differential geometry perspective of shape coherence and curvature evolution by finite-time non-hyperbolic splitting. *arXiv preprint arXiv:1311.5457*, 2013.

[106] Tian Ma and Erik M Bollt. Relatively coherent sets as a hierarchical partition method. *International Journal of Bifurcation and Chaos*, 23(07), 2013.

[107] Hayder Salman, Jan S Hesthaven, Tim Warburton, and George Haller. Predicting transport by lagrangian coherent structures with a high-order method. *Theoretical and Computational Fluid Dynamics*, 21(1):39–58, 2007.

[108] G. Haller and AC Poje. Finite time transport in aperiodic flows. *Physica D: Nonlinear Phenomena*, 119(3-4):352–380, 1998.

[109] G. Haller. Finding finite-time invariant manifolds in two-dimensional velocity fields. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 10:99, 2000.

[110] Francesco d'Ovidio, Vicente Fernández, Emilio Hernández-García, and Cristóbal López. Mixing structures in the mediterranean sea from finite-size lyapunov exponents. *Geophysical Research Letters*, 31(17), 2004.

[111] Illinois institute of technology , available at. `http://mmae.iit.edu/shadden/LCS-tutorial/overview.html`. [Online; accessed 10-05-2010].

[112] S.C. Shadden, J.O. Dabiri, and J.E. Marsden. Lagrangian analysis of fluid transport in empirical vortex ring flows. *Physics of Fluids*, 18:047105, 2006.

[113] Julio M Ottino. *The kinematics of mixing: stretching, chaos, and transport*, volume 3. Cambridge University Press, 1989.

[114] Brock A Mosovsky and James D Meiss. Transport in transitory dynamical systems. *SIAM Journal on Applied Dynamical Systems*, 10(1):35–65, 2011.

[115] Stephen Wiggins. *Chaotic transport in dynamical systems*, volume 2. Springer, 1992.

[116] Hybrid coordinate ocean model (hycom) , available at. `http://www.hycom.org/`. [Online; accessed 06-01-2010].

[117] Rainer Bleck. An oceanic general circulation model framed in hybrid isopycnic-cartesian coordinates. *Ocean modelling*, 4(1):55–88, 2002.

[118] George R Halliwell. Evaluation of vertical coordinate and vertical mixing algorithms in the hybrid-coordinate ocean model (hycom). *Ocean Modelling*, 7(3):285–322, 2004.

[119] NCAR. National center for atmospheric research website dedicated to spill deepwater horizon oil spill. http://www2.ucar.edu/news/ocean-currents-likely-to-carry-oil-spill-along-atlantic-coast, 2010.

[120] Thomas Brox, Christoph Bregler, and Jitendra Malik. Large displacement optical flow. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 41–48. IEEE, 2009.

[121] Thomas Brox and Jitendra Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(3):500–513, 2011.

[122] Stanley Osher Ronald Fedkiw. Level set methods and dynamic implicit surfaces. 2003.