Uncertainty and Error Analysis in the Visualization of Multidimensional and Ensemble Datasets

Dissertation

Presented in Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By

Ayan Biswas, B.E., M.S.

Graduate Program in Computer Science and Engineering

The Ohio State University

2016

Dissertation Committee:

Prof. Han-Wei Shen, Advisor

Prof. Raghu Machiraju

Prof. Huamin Wang

Abstract

Analysis and quantification of uncertainty have become an integral part of the modern day data analysis and visualization frameworks. Varied amounts of uncertainty are introduced throughout the different stages of the visualization pipeline. While visualizing the scientific datasets, it is now imperative to provide an estimation of the associated uncertainty such that the users can readily assess the reliability of the visualization tools. Quantification of uncertainty is non-trivial for scalar datasets and this problem becomes even harder while handling multivariate and vector datasets. In this dissertation, several techniques are presented that identify, utilize and quantify uncertainty for multi-dimensional datasets. These techniques can be broadly classified into two groups: a) analysis of the existence of relationships and features and b) identification and analysis of error in flow visualization tools. The first category of studies uses multivariate and ensemble datasets for analyzing relationship uncertainties. The second category of studies primarily uses vector fields to demonstrate streamlines and stream surface for error analysis.

In the analysis stage, we initially present an information theoretic framework towards the exploration of uncertainty in the relationships of multivariate datasets. We show that, in a multivariate system, variables can show interdependence on each other and information theoretic distance can be effectively used to find a hierarchical grouping of these variables. Using information content as the importance measure, salient variables are identified to start the variable exploration process. Specific mutual information is used for classifying

the isosurfaces of one variable such that they reveal uncertainty regarding the other selected variables. Feedback from the ocean scientists establishes the superiority of this system over the existing techniques. From multivariate relationships, next we discuss the uncertainty in the relationship between ensemble output and input parameters and further propose models for output error estimation. In this case, we show how a spatial and temporal analysis can help in revealing the sensitivity of the input parameters in a multi-resolution ensemble dataset. We employ spatial clustering and temporal aggregation to create an interactive tool for exploration of uncertain sensitivity information. A Bayes' rule-based error estimation approach is provided with another interactive tool for spatiotemporal multi-resolution error exploration. From relationship analysis, we next analyze the uncertainties in feature detection where the feature is a vortex. Vortices are very important features of the flow field but detection of these are not free of uncertainties. Although there are several vortex detection techniques available, they have varying amounts of robustness while detecting these features which in turn introduces uncertainty. Another source of uncertainty is the selection of threshold values for the local point based methods. Here, we use the logistic function to model the threshold selection uncertainty and use multiple uncertain existing detectors to combine them into a more robust vortex detection scheme. Measuring against the domain expert's vortex labels, our proposed method shows higher accuracy compared to the existing vortex methods.

Next, we focus on the visualization tools: streamlines and stream surfaces. For streamlines, we analyze and quantify the error in streamline generation and propose an implicit streamline strategy that scales well with good load balancing. We use a flux-based approach to generate the local streamlines and then use a parallel flux-offset propagation technique

to create a scalar field from a given large two-dimensional vector field. Using this field, isocontour extraction strategy is used for final streamline visualization. This method exhibits much improved performance compared to the existing techniques. Finally, we work with stream surfaces that are popular flow visualization tools and propose four different techniques to quantify the visualization errors. Our proposed techniques provide a trade-off between computation speed and accuracy and we select three popular existing stream surface generation methods to study their behaviors. Using our proposed methods, a comprehensive report is generated to explore how the quality of stream surfaces change depending on the selection of algorithms, and choices of parameters and data complexity.

This is dedicated to my mother Anima Biswas.

Acknowledgments

I would like to acknowledge my advisor Dr. Han-Wei Shen for his help and support throughout my Ph.D. tenure. I thoroughly enjoyed the countless informal and formal technical discussions I had with him over the years. He introduced me to the world of research and guided me through the process of writing conference and journal papers and enhanced my critical thinking. I feel privileged to be part of his research team and thankful for his advice.

I would also like to thank my dissertation committee members, Dr. Raghu Machiraju and Dr. Huamin Wang. Their valuable comments and suggestions have improved my research work considerably. I would also like to thank Dr. Yusu Wang for being a part of my candidacy examination committee. Further, I would like to extend a big thanks to the visualization group of Los Alamos National Laboratory (LANL). I thank Dr. Jonathan Woodring for mentoring me in those summer internships at LANL. I also got valuable suggestions and help from Dr. James Ahrens, Richard Strelitz, John Patchett, Francesca Samsel and all others from the visualization group of LANL.

Next, I would like to thank all my GRAVITY lab-mates and friends Chun-Ming Chen, Xiaotong Liu, Soumya Dutta, Wenbin He, Tzu-Hsuan Wei for their suggestions and selfless help. I would like to mention the names of my other friends from that lab including Xin Tong, Kewei Lu, Subhashis Hazarika, Cheng Li, Ko-Chih Wang, Junpeng Wang. With all of you, my Ph.D. years were truly memorable that I will remember forever.

I would like to extend a special thanks to Dr. David Thompson and Dr. Guang Lin for providing me with their domain knowledge. The research discussions with you were truly a learning experience and opened new avenues of research for me. Thank you for your time and patience.

Finally, I would like to thank all my family members who are back in India and always praying for my success. My unconditional gratitude goes towards my parents without whom I would not be here. They have supported and motivated me over all these years. I would also like to thank my uncles and aunts, and my little cousins. I miss my uncle (Late) Mr. Satyen Mitra who would be the happiest to see me with my doctoral degree. I would also like to express my regards to my parents-in-law for their love and concern. And then, two years back, I found my best friend and soul mate with whom I am going to share the rest of my life. Thank you Suranjana for being the best wife in the world and putting up with me in those good and hard times.

Vita

2007	B.E. Computer Science and Engineering, Jadavpur University, Kolkata, India
2007–2010	Software Engineer, STMicroelectronics, India
2010–2011	Graduate Teaching Assistant, The Ohio State University
2011–present	Graduate Research Assistant, The Ohio State University
2015	M.S. Computer Science and Engineering, The Ohio State University
May–Aug, 2012–2015	Summer Research Intern, Los Alamos National Laboratory

Publications

Research Publications

Ayan Biswas, Soumya Dutta, Han-Wei Shen, and Jonathan Woodring "An Information-aware framework for Exploring Multivariate Data Sets". *IEEE Transactions on Visualization and Computer Graphics and IEEE Visualization Conference*, October, 2013.

Ayan Biswas and Han-Wei Shen "Evaluation of Stream Surfaces Using Error Quantification Metrics". *SPIE VDA 2014*, February, 2014.

Ayan Biswas, David Thompson, Wenbin He, Qi Deng, Chun-Ming Chen, Han-Wei Shen, Raghu Machiraju, and Anand Rangarajan "An Uncertainty-Driven Approach to Vortex Analysis Using Oracle Consensus and Spatial Proximity". *IEEE Pacific Visualization 2015*, April, 2015.

Ayan Biswas, Richard Strelitz, Jonathan Woodring, Chun-Ming Chen, and Han-Wei Shen "A Scalable Streamline Generation Algorithm Via Flux-Based Isocontour Extraction". *Euro Graphics PGV* , June, 2016.

Ayan Biswas, Guang Lin, Xiaotong Liu, and Han-Wei Shen "Visualization of Time-Varying Weather Ensembles Across Multiple Resolutions". *IEEE Transactions on Visualization and Computer Graphics and IEEE Visualization Conference*, October, 2016.

Chun-Ming Chen, Ayan Biswas, and Han-Wei Shen "Uncertainty Modeling and Error Reduction for Pathline Computation in Time-varying Flow Fields". *IEEE Pacific Visualization 2015*, April, 2015.

Yu Su, Gagan Agarwal, Jonathan Woodring, Ayan Biswas, and Han-Wei Shen "Supporting Correlation Analysis on Scientific Datasets in Parallel and Distributed Settings". *HPDC 2014*, 2014.

Tzu-Hsuan Wei, Chun-Ming Chen, and Ayan Biswas "Efficient Local Histogram Searching via Bitmap Indexing". *EuroVis 2015*, May, 2015.

Fields of Study

Major Field: Computer Science and Engineering

Studies in:

Computer Graphics	Prof. Han-Wei Shen
High Performance Computing	Prof. P. Sadayappan
Theory and Algorithms	Prof. Tamal Dey

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Background and Motivation

Visualization has been established as a powerful tool for data analysis due to its acceptability to a broader audience. Most of the visualization algorithms were previously developed based on the assumption that the datasets under consideration are fully accurate and the generated visual outputs were completely reliable. But this does not quite hold true in reality since there are myriads of factors that introduce error and uncertainty throughout the different stages of visualization process. For creating a visualization output that represents a physical phenomenon, we need to pass through data acquisition to data transformation to data analysis before applying visualization algorithms and generating a visual output. In this visualization pipeline, as presented in Figure 1.1, every stage generates uncertainty which is added up throughout the pipeline. Since visualization is used to make sense of the data and take decisions based on what is observed on the screen, it is imperative to get an understanding of the reliability of the visual output. In the past decade, researchers have invested a considerable amount of effort in establishing uncertainty and error analysis as an integral part of visualization process.

Understanding the different kinds of uncertainty is the first step towards creating a meaningful uncertainty analysis framework. Uncertainty encountered at the "Analysis"

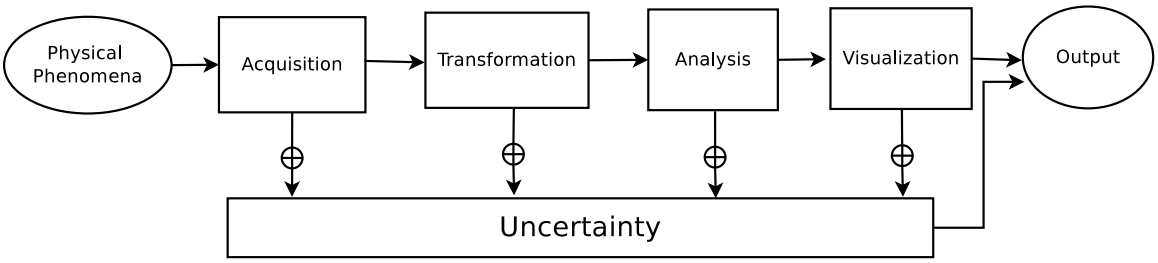


Figure 1.1: Overview of the visualization pipeline.

stage is thus different from the uncertainty in the "Visualization" stage. In the "Analysis" stage, generally the goal is to determine or estimate the existence of a feature (e.g., vortex) or a relationship (e.g., how one variable changes when another variable is altered). This uncertainty can be attributed to vague definition of features, error or uncertainty introduced in the acquisition stage, or lack of full domain knowledge etc. Compared to this, the reliability of the algorithm generated outputs (e.g., streamlines, stream surfaces) are analyzed in the "Visualization" stage. The source of this uncertainty is generally the particular algorithm used for generating the visualization among all the available options. For example, there are multiple stream surface generation algorithms proposed which all generate similar but slightly varying visualization results. It is also critical to analyze and depict this uncertainty to the user.

This dissertation describes approaches towards identifying and analyzing the uncertainties at the algorithm level ("Analysis" stage of the visualization pipeline) and the feature level ("Visualization" stage of the pipeline) in the visualization pipeline and enhance the understanding of the dataset. In the first three chapters, the research focus is on the existence of relationships and features. Multivariate datasets are very common in applications

of computation sciences, and scientists want to explore the relationships among the different variables of the system. In these scientific simulations, the total number of variables produced can be quite large and the exploration can become tedious and complex. Providing methodical and systematic steps for exploration of such datasets is thus an important aspect of data visualization. Similarly, ensemble datasets have become more popular recently since computers are now fast enough to run the simulations multiple times with varying input parameters and generate a sequence of results at different time steps and resolutions. Exploring such relationships between the input parameters and the ensemble output is of utmost importance to understand the important variables of the system. Further moving on to the uncertain features, robust identification of vortex-like features have been historically very important. Although there exist multiple vortex detection methods, none of these can yield complete prediction accuracy. Thus, understanding the different reasons of uncertainty and utilizing them to form a robust detector has been an ongoing challenge.

The next part of the dissertation concentrates on the error computation and visualization in the streamline and stream surfaces of vector fields. Streamlines are very popular tools for flow visualization and parallel streamline methods are plentiful. Most of these methods parallelize the streamlines across blocks or over seeds, but none of these algorithms try to parallelize the computation of the individual streamlines for speed up. A parallel-over-segments streamline algorithm that is accurate and has the load-balancing and scalability properties, has remained largely unexplored. Similarly, for stream surfaces, there exist multiple algorithms which all produce similar but slightly varying stream surfaces given the same vector field. Since there is little existing work on the computation of the errors on these surfaces, understanding the reliability of these visualizations is not possible. Also,

there is a need for such quality measures on the stream surfaces when we want to select one algorithm over the other ones.

1.2 Research Problems

In this dissertation, we aim at answering specific questions from the field of data analysis and visualization. These questions include:

1. How to effectively explore multivariate datasets when the users may or may not be familiar with the system variables and properties? Since multivariate datasets are prevalent, can there be a framework that provides guidance to the users by offering suggestions at each stage of the exploration process? How to integrate and incorporate the notion of multivariate uncertainty via simple and intuitive visualizations to help the scientists in the exploration process?

2. How to explore and understand the sensitivity of an ensemble simulation where multiple inputs are available? Given the set of input parameters and the ensemble output, which parameters are most important in deciding the output? How the sensitivity of the individual parameters change as a function of spatial location, time step and across different resolutions? In the presence of the ground truth, how does the prediction accuracy of the ensemble runs change for different resolutions? Does the higher resolution always have higher accuracy?

3. How to provide robust feature detection and analysis when the feature under consideration contains uncertainty? Since vortices are utmost important features of the vector field and contain multiple sources of uncertainty, how to identify and utilize those

sources of uncertainties to improve the detection results? Is it possible to enhance the results by using multiple feature detectors and domain expert's pre-classified labels?

4. How to generate a large set of streamlines in a load-balanced and scalable technique without reducing the quality of the integral curves? Since streamlines are very popular for visualizing vector fields and multiple techniques for parallelizing the computation exist, can we get further performance improvement if individual segments of a streamline are generated in parallel and stitched accurately?

5. How to quantify and visualize the error encountered in the stream surface generation process? Is it possible to identify the different sources of error in this visualization? How to compare the multiple existing stream surface algorithms based on their accuracy and time complexity trade-off?

1.3 Contributions

This dissertation primarily contributes to the areas of uncertainty and error analysis for variable and parameter analysis, feature detection and feature visualization. Specifically, this dissertation covers (a) Analysis of relationship uncertainties of multivariate system, (b) Analysis of the relationship between input parameters and ensemble output with output error analysis, (c) Analysis of existence of an uncertain feature using domain experts inputs, (d) Error-aware generation of integral curves and surfaces.

First, this dissertation provides a new framework of multivariate data exploration system. This framework

1. Guides the users at each step of the exploration process by providing an intuitive and interactive interface.

2. Uses information theory to quantify saliency and uncertainty remaining in the system.

3. Incorporates specific mutual information methods to classify the iscontours of variables based on the relationship of one variable with the others to provide effective simultaneous visualizations.

4. Uses a very easy-to-use graph-based approach to cluster a system of variables and analyze the interaction among the clusters of variables and withing the clusters.

5. Has been used by the domain experts for the detailed exploration of real world datasets.

Second, this dissertation further looks into the relationship uncertainties of the ensemble output and the input parameters of a simulation and analyzes error in the output when the ensemble dataset can be both time-varying and multi-resolution. This research

1. Provides a thorough and in-depth sensitivity analysis of ensemble datasets. This analysis spans across spatial and temporal domains and extends to multiple resolutions of such ensemble datasets.

2. Generates visualizations that enable the domain experts to quickly understand the non-trivial sensitivity relationships and how these change when space or time or resolution of the data changes.

3. Utilizes an effective and intuitive interface for selection of parameters, variables and spatial locations for quick and easy interaction from the users.

4. Analyzes and quantifies the predictive uncertainties of the ensemble outputs for a detailed error estimation with the ground truth available.

5. Provides a complete workflow to handle spatiotemporal analysis of the multi-resolution ensemble datasets with the domain expert in the exploration loop.

Third, we contribute to the hard problem of uncertain feature detection where vortices have been discussed as the intended feature. This research

1. Identifies and analyzes the multiple sources of uncertainty in the vortex detection scenario.

2. Introduces a new approach towards understanding the existing vortex detectors by using their threshold values to create a fuzzy input-based system.

3. Presents a consensus-based voting method to identify the different regions according to their varying uncertainty level.

4. Introduces the use of spatial proximity into the vortex detection framework to enhance and refine the detection process.

5. Provides a workflow of uncertain vortex detection by using the domain expert's knowledge to determine the important parameters of the system and verify the final result with the experts to establish the performance improvement.

Fourth, this dissertation further contributes towards error analysis and quantification of integral curves and surfaces for vector fields. This body of research

1. Addresses the error-aware generation of streamlines for very large scale two-dimensional datasets.

2. Provides fast streamline generation by using a parallel flux-based implicit streamline generation method.

3. Proposes a novel streamline stitching algorithm seamlessly in parallel across local regions.

4. Generates a multi-processor workflow to achieve speed up, scalability and load balancing.

5. Introduces four distinct error quantification methods for the stream surfaces to facilitate identification of regions of varying uncertainty of these surfaces.

6. Creates a workflow for comparing existing stream surface generation and visualization algorithms in terms of their speed and accuracy trade-off.

7. Further extends the understanding of the stream surface algorithms by exploring their parameter selection and their effect on the quality of the generated surfaces.

1.4 Outline

In this dissertation, we break our research into five main chapters from Chapter 3 to Chapter 7. Chapter 2 provides the background and literature related to our research chapters. Chapter 8 provides the conclusions and future research opportunities. We provide a brief summary of the research chapters below.

1.4.1 Uncertainty Guided Isocontour Selection for Exploration of Multivariate Relationships

Multivariate datasets are common in scientific visualization since many real-world phenomena are dependent on multiple factors. Exploration of the non-trivial relationships among the variables of a multivariate system is an important task to the experts. For a univariate system, isocontours are popular tools for showing the regions of constant scalar. For

multivariate datasets, isocontours of one variable can be used to explore the relationships with another variable by the showing that variable as a color on that isocontour. This idea enables us to define multivariate uncertainty and uses information theory for identification of the interesting isocontours in the multivariate context. In depth discussions on this is presented in Chapter 3.

1.4.2 Sensitivity and Accuracy Analysis on Time-varying Ensemble Data Across Multiple Resolutions

Similar to the multivariate relationships, given a set of input parameters for a time-varying ensemble dataset, identification of the input-output relationships (i.e., sensitivity) is also non-trivial. With the increase in the computational power, scientists now generate multiple resolutions of these datasets which make the analysis even harder. It is of utmost importance to the domain experts to understand the effect of the input parameters on the output and identify the salient ones. Since many of the parameters used in the simulations are non-physical and there exist non-linear input-output relationships, understanding how the sensitivity of the parameters change as a function of space and time across resolutions is of prime importance. Along with this, when ground truth is available with the ensemble datasets, analyzing the accuracy is also needed. In Chapter 4, we provide tools and techniques for the spatiotemporal analysis of sensitivity along with an accuracy analysis method for multi-resolution ensemble datasets.

1.4.3 Uncertainty-driven Vortex Detection in Flow Fields

Along with the exploration of relationships of variables and input parameters, analysis of the existence of features is also a heavily researched topic. For a vector field, vortices are such features that have attracted much attention due to their complexity in the detection

process. Vortex detection process is not free from errors, mainly because there is no mathematical definition of what describes a vortex. Over the years, scientists have come up with different methods but all of these methods are known to produce false positives and false negatives when investigated by domain scientists. In a vortex detection process, one source of uncertainty is the inconsistency in the outcome of the existing vortex detectors. For a given region, different vortex methods can have disagreements in predicting whether that region should be part of a vortex structure or not. Point based vortex detectors are popular although these detectors have another source of uncertainty. These methods generally use a threshold above which every value is considered a vortex. But if the detector's output is close to the threshold, the prediction is less certain. We investigate these two sources of uncertainty and combine a set of existing vortex identification methods to increase the certainty in the prediction. This is discussed in Chapter 5.

1.4.4 Error-aware Flux-based Algorithm for Scalable Streamline Generation

After analysis of relationships and features in the previous chapters, we focus on the uncertainty and error analysis in the visualization algorithms for the vector fields. Specifically, we concentrate on the streamlines (Chapter 6) and stream surface generation algorithms (Chapter 7). Streamline is a popular flow visualization tool as it is intuitive and a collection of these curves can depict the flow structures well. But streamline can contain uncertainties depending on what algorithm and which parameters are used to generate it. Nowadays, the vector datasets we deal with are quite large and there exist multiple parallel streamline generation algorithms. Still generation of large amounts of streamlines with a scalable algorithm without losing the visualization accuracy is a big challenge. Chapter 6

discusses a parallel implicit streamline generation method and elaborates its performance and accuracy in detail.

1.4.5 Error Measurement on Stream Surfaces

Stream surfaces are also extensively used for visualizing flow fields since they can provide better depth cues compared to streamlines. There exist different algorithms for generating stream surfaces and these algorithms use different sets of parameters. While visually exploring flow fields using stream surfaces, it is important to provide an estimation of quality although it is mostly lacking in the literature. We conduct our research to propose techniques that can quantify the amount of error incurred on the stream surfaces and visualize them. We propose four different techniques for error estimation on the stream surfaces and compare different existing algorithms. We formulate a thorough study of how these algorithms behave when the different parameters are altered and different datasets are selected. These are presented in detail in Chapter 7.

Chapter 2: Related Work

This chapter reviews existing publications that are relevant to the following chapters. They are provided in no particular order although some consistency with the chapter ordering is preserved.

2.1 Multivariate Data Analysis

Multivariate data analysis and visualization is an active research area and its applications can be found across many fields. For a comprehensive study of this topic, reader can refer to the survey works by de Oliveira et al. [33] and Wong et al. [144]. An analysis guided multivariate exploration was proposed by Yang et al. [150] where valuable information nuggets were first extracted based on user's interest and exploration continued with the discovery of similar nuggets. Martin and Ward [87] incorporated easy brushing of high dimensional data to improve the interactivity of the exploration system. Jänicke et al. [70] later facilitated the exploration of high dimensional data in two-dimensional space by applying a transformation of the high dimensional data attributes to point clouds. To handle extremely large multivariate datasets, a new visualization system was proposed by Rubel et al. [110]. Gosink et al. [48] used a statistics-based framework to adapt the existing query-driven visualization framework for large multivariate analysis. In another work, flexible linked axes were used by Claessen and Van Wijk [28] for multivariate data

visualization. Guo et al. [53] used PCP and MDS based projection techniques to facilitate transfer function design. Recently, Wang et al. [138] used transfer entropy to explore causal relationships in a time-varying multivariate dataset. PCP [67, 66, 68] is a very popular tool for multivariate data exploration although visual cluttering can pose problems in the exploration of variable relationships. Removal of visual clutter [75, 32] and ordering of neighboring variables axes [5, 100, 85, 63] remain an ongoing research topic.

2.2 Applications of Information Theory in Visualization

Information theory [29] has been a widely used tool for solving a variety of problems in visualization and graphics field. Gumhold [51] used entropy to decide the placement of light sources when camera parameters are varied. Many view point selection algorithms [136, 14, 137] have been proposed that are based on information-theoretic measures. Scene visibility and radiosity complexity were also analyzed by Feixas et al. [42]. An importance driven approach was proposed by Wang et al. [139] where they applied a block-wise analysis on volumetric time-varying data. An information-theoretic framework for flow field data exploration was provided by Xu et al. [147]. Chen and Jänicke [26] considered visualization as a visual communication channel and used information theoretic measures to compute the effectiveness of a given visualization.

2.3 Selection of Salient Isocontours

Isosurfaces have been extensively studied in the past for the scalar datasets. Selection of isovalues is important for scalar dataset exploration and previously a dataset's statistical properties [6, 39, 117] were popularly used as a measure of saliency of isocontours. Also the topology of isosurfaces have been used [23, 155] to gain understanding of a scalar

dataset. In more recent works, geometric properties of isosurfaces such as fractal dimensions [80] and distance transforms [18] have also been used for determining the saliency of isosurfaces in scalar datasets.

2.4 Uncertainty Analysis for Ensemble Datasets

Simulations generating ensemble outputs are the main source of uncertain datasets. Use of ensemble datasets is quite common for the weather analysts and visual exploration of the uncertainty related to these ensemble datasets has been a popular topic of research. A framework called Ensemble-Vis was created by Potter et al. [105] for weather forecasting and climate modeling where they combined multiple ensemble visualization techniques. Sanyal et al. [114] proposed Noodles as an enhancement to the existing Spaghetti plots for understanding the meteorological datasets. They used circular glyphs and confidence ribbons to depict the Euclidean spread of isocontour lines effectively. Recently, Hao et al. [57] proposed a kd-tree based exploration technique for temporal particle ensemble datasets. Another visual verification system was proposed by Bock et al. [13] to explore the Coronal Mass Ejection phenomenon. Demir et al. [34] extended the bar and line charts to design Multi-charts that facilitated the comparative exploration of multiple volume datasets. They used bidirectional brushing and linking to facilitate user interaction and query visualization. For providing a comparative visualization of 2D temporal ensemble datasets, Höllt et al. [58, 64] introduced time-series glyphs. Other notable comparative visual analysis methods for vector ensembles were proposed by Guo et al. [52] and Jarema et al. [71]. Summarization methods for ensemble space curves were discussed in the works of Whitaker et al. [143], Mirzargar et al. [91] and Ferstl et al. [43]. Bensema et al. [9]

introduced modality-based classification for the high variance regions of ensemble datasets to facilitate a detailed understanding of the ensemble distributions.

2.5 Sensitivity Analysis

The goal of sensitivity analysis is to identify the most influential parameters from a set of uncertain input parameters for a simulation. This sensitivity study helps the scientists to understand and verify their simulation model, prioritize the parameters for more effective sampling of parameter space, and even simplify the simulation model. The existing measures for sensitivity analysis can be broadly classified into two groups: local and global [69]. The local methods aim at observing the change in the output when the input parameters go through small perturbations. These methods generally compute partial derivatives at important feature points e.g. around the mean. The local methods were popular in the earlier phase of sensitivity analysis works [21, 65, 76] and also have been applied for hydrological modeling recently [24]. Compared to the local techniques, global methods compute the sensitivity by varying the parameters throughout the possible distribution range [125] and are often preferred over the local counterparts [69]. A variance-based global sensitivity quantification method was proposed by Sobol [127]. Saltelli et al. [112] proposed a Fourier amplitude sensitivity test (FAST) for finding the total influence of individual parameters on the output. A moment-independent sensitivity measure was given by Borgonovo [15] that computed the sensitivity of parameters by observing the change in the output distribution. Among other popular methods, the derivative based global sensitivity measure (DGSM) by Sobol and Kucherenko [126], incremental ratio based input parameter screening method by Morris [92] are the most popular.

2.6 Vortex Detection Methods

Depending on the neighborhood of the data upon which the computation of the vortex detectors is performed, these detectors can be classified as either local or global [130].

2.6.1 Local Detectors

Local vortex detectors generally use the jacobian of the velocity field to determine whether the individual points are part of a vortex structure. Few popular examples of such local jacobian based detectors are the Q-criterion [62], the λ_2-method [72], the Δ-criterion [27], and the Γ_2 method [49]. The performance of these detectors vary with the complexity of the datasets and generally there is no consensus on which is the best detector [145] although λ_2 method is widely used for its consistency [116]. Roth and Peikert [109] introduced the parallel vector operator to locate the core lines of the vortices. The second derivative of the velocity field was employed by the authors to detect slowly rotating vortices. Other approaches based on parallel vectors [99, 108] also exist.

2.6.2 Global Detectors

Global vortex detection methods often employ streamlines and/or pathlines to identify regions of vortical structures. A streamline-based winding angle method was effectively used by Sadarjoen *et al*. [111] for vortex detection. Another streamline-based method was proposed by Jiang *et al*. [73] where they analyzed the geometry of the streamlines and detect vortex core regions. A predictor and corrector method was proposed by Banks and Singer [7] for extraction of vortex core lines. An objective definition of vortices in the rotating reference frame was provided by Haller [56]. His method was similar to that of Q-criterion except more global information was incorporated in the detection process via a

Lagrangian analysis. A good survey of the vortex detection methods was provided by Jiang *et al*. [74]. Certain local geometric properties of streamlines were computed by Pagendarm *et al*. [97] to form a model vortex for curvature density field estimation. This density field was later used to vortex core identification with the help of isosurfaces. Recently, a semi-automatic technique was employed by Kohler *et al*. [82] where λ_2-method and line predicates were used for blood flow visualization.

2.7 Streamline Generation Algorithms

Integral curves are very popular for analyzing vector data and it has been widely used in the past. A comprehensive discussion on the basics of streamlines and related issues can be found in the survey by McLoughlin *et al.* [89]. Here we briefly point out the implicit streamline and existing parallel streamline techniques that are closely related to our proposed method.

2.7.1 Implicit Streamlines

Although the use of numerical integration is popular for generating integral curves, it can be problematic depending on multiple factors [79]: method of integration, selected step size, interpolation scheme etc. Implicit integral curves and surfaces are an alternative to numerical integration based methods which mostly stem from the idea of stream functions that are popular for two-dimensional incompressible flows as well as three-dimensional flows [46, 152, 50, 77, 41, 88, 122, 123] since a long time. Some notable works in this topic were provided by Kenwright and Mallinson [79, 81], Beale [8], Van Wijk [135]. A closely related concept is the analysis of vector field using the popular Helmholtz-Hodge decomposition [103, 131]. Although widely popular, implicit streamline generation using

these methods was mostly limited to serial sequential algorithms. In this work, we take up the idea of implicit streamlines and extend it to work efficiently in parallel environment.

2.7.2 Parallel Streamlines

Due to the complexity of particle tracing and its data dependency, how to compute streamlines in parallel has been an important research topic. To compute streamlines on very large datasets in a distributed setting, Camp *et al.* [22] study the benefits of parallelize-over-seeds and parallelize-over-blocks, and propose a hybrid approach. Peterka *et al.* [102] study data partitioning and job assignment for the parallelize-over-blocks scenario and find that the Round-Robin partitioning generally achieves better load balance. Nouanesengsy *et al.* [95] use a preprocessed graph representation of the flow field to optimize block assignment to the processes. Muller *et al.* [93] take the dynamic job assignment approach to balance the workloads by a work requesting algorithm for the parallelize-over-seeds scheme. Kendall *et al.* [78] propose a MapReduce-like framework called DStep to achieve high scalability. Recently, Agranovsky *et al.* [1] provided an interpolation-based integral curve generation method for performance and accuracy improvement.

2.8 Stream Surface Algorithms

Stream surfaces are well known to the visualization community since the early days of flow visualization. The first most popular stream surface algorithm was given by Hultquist [61]. This simple algorithm generated a triangulation of neighboring streamlines and streamline seeds were inserted or deleted depending on their divergence or convergence. Garth et al. [45] improved this algorithm by applying arc length parameterization and by taking into account the curvature of the flow for seed insertion. Comparing to these explicit stream surface algorithms, there exist implicit algorithms. Van Wijk [135] first proposed

such implicit stream surface algorithm which was later improved upon by Stöter et al.[129]. Schafhitzel et al. [115] and McLoughlin et al. [90] proposed stream surface algorithms that use less complex data structures. Other stream surface methods were also proposed by Scheuermann et al. [118] and Laramee et al. [83]. A good discussion of surface based flow visualization techniques can be found in the survey work by Edmunds et al. [40]. Recently, Schulze et al. [121] have proposed a method for generating a well-shaped stream surface mesh formation by forcing the front line to the perpendicular to the flow.

2.9 Uncertainty in Vortex and Stream Surfaces

Uncertainty analysis and quantification have become an integral part of flow visualization. In a survey work, Pang et al. [98] have given a detailed description of the sources of errors and presented several uncertainty based flow visualization techniques. Recently, Pöthkow et al. [104] have also provided a comprehensive list of references about uncertainty and error visualization techniques. For vortex detection, Otto and Theisel proposed a method [96] where they assumed that the points of dataset followed a Gaussian distribution. They applied Gaussian fitting and Monte-Carlo based approach to provide an uncertainty based vortex detection method. Previously, Burger *et al*. [20] incorporated fuzzy vortex detectors for intuitive visual exploration of vortices. In the context of stream surfaces, the error quantification measures are also mostly lacking. Garth et al. [44] used a set of streamlines as the ground truth to compare the quality of the generated stream surfaces. Schneider et al. [119] investigated the accurate creation stream surface when it approached a critical point.

Chapter 3: Uncertainty Guided Isosurface Selection in Multivariate Data Exploration

Exploration of multivariate datasets is an integral part of scientific visualization as in most real world phenomena, there exist multiple factors associated with the complex interactions of different variables. To gain an in-depth understanding of a scientific process, the relationships among the variables needs to be thoroughly investigated. However, exploration of several variables simultaneously can be both tedious and confusing. Identification of informative isocontours is an important aspect of univariate as well as multivariate data exploration. For univariate datasets, isocontours reveal the regions of the same scalar value. Given a scalar dataset, identification of salient isovalues has attracted much attention from the researchers over the years. Previously, saliency of isosurfaces was determined based on the shape [6, 39, 117], topology [23, 155], and geometry [80] of these contours. Identification of salient isocontours is an even harder problem as we move on from univariate system to multivariate datasets. For a multivariate system, as there can be interdependences among multiple variables, isocontours of selected variables can reveal information about the associated variables and how they interact. Since, isocontour of one variable shows the spatial distribution of a selected value of one variable, the distribution of values of other variables on that isocontour provides a notion of "uncertainty" of that other variable. Understanding the behavior of one variable when a related variable remains constant, is a

powerful means of exploring the relationship between these two variables. In this chapter, we explore how information theoretic measures can be applied to facilitate the selection of informative isocontours and help us quantify the uncertainty across related variables.

This chapter introduces an information-aware framework to help the users in multivariate data exploration. Previously, information theory based tools have been used in visualization and graphics field to solve a variety of problems and in this framework, a similar approach is adopted. For uncertainty guided isocontour selection, the mutual information among the variables is decomposed into specific information metrics and an interactive interface is provided to facilitate the isovalue selection process. Since this measure takes advantage of information overlap between variables, we make use of the mutual information to create subgroups of the variables according to their information overlap. Inside the subgroup, since not all the variables involved in that subsystem contribute equally towards the joint entropy (or the total information content) of the subgroup, conditional entropy measure is used to calculate the relative importance of the variables. After the individual variables are selected, to explore the relationship among the variables, the specific information measure is used to calculate the informativeness of each scalar value based on how it is related to the values of other variables. Isocontours of the selected variables can be identified which provide the uncertainty information about the other variables at the same locations. For the ease of data exploration, we designed an intuitive interface that encapsulates our proposed exploration framework and presented it to the domain experts for their feedback. Domain experts used the system on their ocean dataset and provided encouraging feedback regarding our system.

Our contributions in this chapter are threefold:

1. We use specific information to classify the isocontours of variables based on the relationship of one variable with the others, and effectively provide simultaneous visualization of the related variables.

2. We use a novel graph-based approach to analyze and cluster a system of variables and analyze the interaction among the clusters and within the clusters.

3. We introduce a new framework for multivariate data exploration which guides the users at each step of the exploration process by providing an intuitive and interactive interface.

3.1 Specific Information Based Isovalue Selection

In our information-theoretic framework, we take advantage of two specific information measures, Surprise and Predictability, that are derived from mutual information. In this section, we review the concepts of these measures and discuss their usefulness in a multivariate framework.

3.1.1 Definitions

Entropy

Information theory provides us with a method for quantifying the information content of a random variable by using Shannon's entropy calculation. For a random variable X, Shannon's entropy $H(X)$ is defined as

$$H(X) = -\sum_{x \in X} p(x) \log p(x). \tag{3.1}$$

This is a measure of the uncertainty about the given random variable. For univariate datasets, Shannon's entropy has been previously used to quantify the uncertainty of variables. The histogram of a variable is used as the probability mass function to calculate the total uncertainty of that variable using Equation 3.1.

Mutual Information

In information theory, mutual information between two random variables is the measure of information overlap or the correlation between the variables. For two random variables X and Y, mutual information $I(X,Y)$ is defined as

$$I(X,Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}. \tag{3.2}$$

As a correlation metric, mutual information has an advantage over the correlation coefficient metric as it can measure non-linear relationships as well. Mutual information between two variables measures the informativeness of one variable about the other variable.

Mutual information estimates the informativeness of one variable against another variable. It is also possible to measure the information associated with a specific scalar value x, $x \in X$, about another random variable Y, which is termed as *specific information*. In this case, the variable X is called the *reference variable*. We utilize the specific information of the reference variable to identify salient isocontours, where the saliency is defined by how much the uncertainty of one variable is reduced after observing an isocontour from the reference variable. There exist several ways to calculate specific information, and we utilized two of them that were introduced in the works of DeWeese and Meister [36]. These two specific information metrics were named as Surprise and Predictability by Bramon et al. [17], all based on a decomposition of the standard mutual information.

Surprise

Surprise, which is also referred to as I_1, was introduced and analyzed by DeWeese and Meister [36] and it is given as:

$$I_1(x;Y) = \sum_{y \in Y} p(y|x) \log \frac{p(y|x)}{p(y)}. \tag{3.3}$$

$I_1(x;Y)$ is always positive as it represents the Kullback-Leibler distance between $p(Y|x)$ and $p(Y)$. A high $I_1(x;Y)$ value indicates that some infrequent occurrences $y \in Y$ have become more probable due to the observation of x which amounts to a surprising result, where x is a value of the reference variable X. The values of x, for which $I_1(x;Y)$ is high, are representative of the isovalues which are of interest to us.

Predictability

DeWeese and Meister also introduced the metric Predictability or I_2 which is given as:

$$\begin{aligned} I_2(x;Y) &= H(Y) - H(Y|x) \\ &= -\sum_{y \in Y} p(y) \log p(y) + \sum_{y \in Y} p(y|x) \log p(y|x) \end{aligned} \tag{3.4}$$

Here $H(Y)$ refers to the entropy of variable Y as presented in Equation 3.1. $I_2(x;Y)$ gives the amount of reduction in uncertainty about Y after observing the data value x. It is to be noted that, unlike I_1, I_2 can take negative values which suggests that there are certain observations x for which our uncertainty about Y may increase. The values of x, for which $I_2(x;Y)$ is high, are representative isovalues of the reference variable X that reduce the uncertainty about Y and are of interest to us. On the other hand, the values of x, for which $I_2(x;Y)$ is low or negative, represent a high uncertainty about the variable Y at the location of the isocontours, which also prompts us to perform further exploration.

3.1.2 Properties of Specific Information Measures

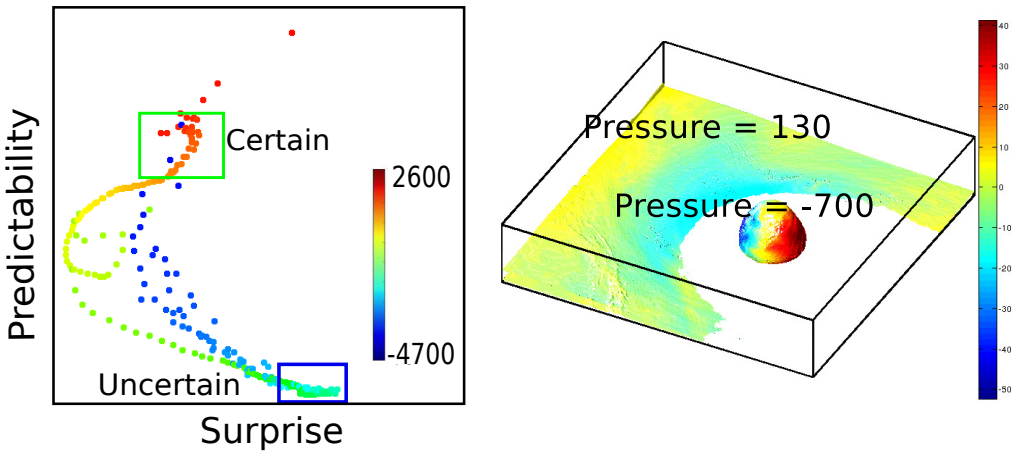
Between the two specific information measures, Predictability or I_2 is an additive metric. This additive property establishes that when I_2 is measured based on two observations x and y, it amounts to the sum of I_2 calculated based on x, and I_2 based on y given that x is already known.

$$I_2(x,y;Z) = I_2(x;Z) + I_2(y;Z|x).$$

As additivity is a desirable property of a metric, Bramon et al. [17] and DeWeese and Meister [36] have considered I_2 to be more intuitive as a measure of specific information. In our work, we utilize both I_1 and I_2 for the selection of interesting scalars.

These specific information measures provide us with the tools for classifying the individual scalars of a variable. Given two variables of a dataset, each of them can be considered as a random variable and for each scalar value of the reference variable chosen between the two variables, the I_1 and I_2 metrics can be computed. To guide the exploration of the dataset, the scalars which have high I_1 value are identified as the surprising ones and are further classified by their I_2 values. If a surprising value has higher predictability, then the corresponding isocontour of the variable will reflect a confident state of the other variable. Conversely, if the predictability is low, then the scalar value associated with the reference variable will not be able to predict the state of the other variable with high certainty.

An illustrative example is provided in Figure 3.1. Here, the Isabel Hurricane dataset is used and the two variables used for exploration are Pressure (X) and U-Velocity (Y). As shown in Figure 3.1(a), the $I_1(x;Y)$ and $I_2(x;Y)$, $x \in X$, are computed and a 2D scatter plot representing mapping of the scalars of X to the I_1I_2 space is generated. In this figure, each point represents an (I_1, I_2) pair computed from a specific scalar value of Pressure (X). In



(a) The I_1I_2 map of the user selected variables, Pressure and U-Velocity.

(b) Generated isosurfaces for the selected isovalue of the I_1I_2 map.

Figure 3.1: An example of isovalue selection from the Isabel Hurricane dataset.

this map, different scalars of X are color mapped according to their value which makes it easier for users to see any relationship patterns. Users are provided with an option to select a point or a region from the I_1I_2 scatter plot with a rectangular window, and the corresponding scalar values are selected. The selected values are then used to draw the isosurfaces on the variable X which are color mapped by the scalars of Y as shown in Figure 3.1(b). To identify the regions where the isosurface of X faithfully represents the value of Y, the higher I_1 and higher I_2 values are selected. Conversely, to identify the regions where the isosurface of X has high variation in values of Y, the higher I_1 and lower I_2 values are needed to be selected. The green rectangular region selects the scalars from the Pressure field that have less variability in U-velocity values. Conversely, the blue region selects the values which have much more variability. Figure 3.1b shows the two examples of isosurfaces as a result of user's selection. The isosurface corresponding to Pressure value -700 comes from the blue rectangular region and it has high variability in the U-velocity.

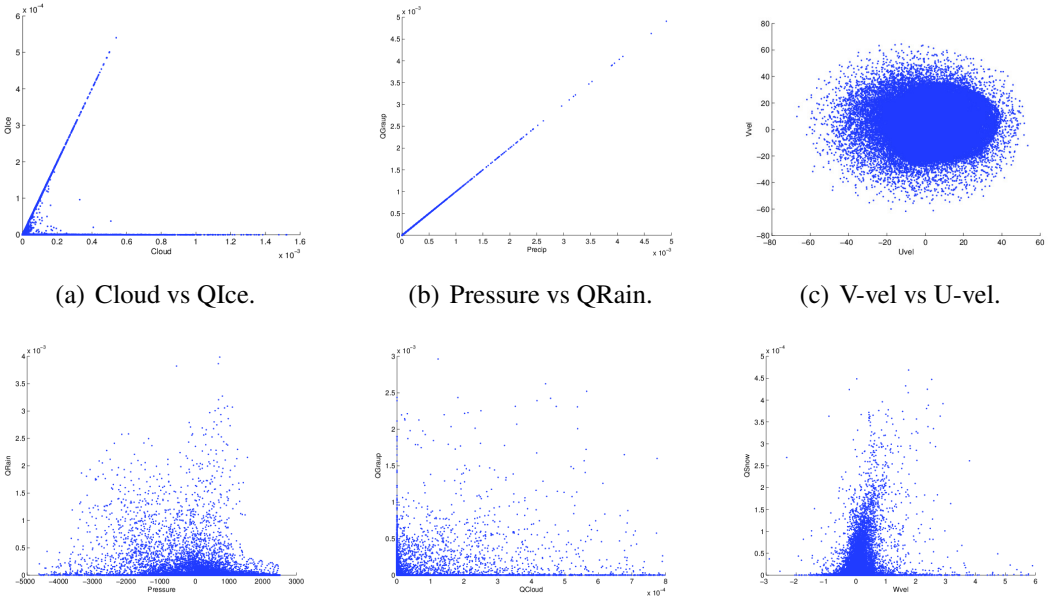
The isosurface for Pressure value 130 is selected from the green rectangular region has much less variability in U-velocity.

3.2 Selection of Variables for Uncertainty-Based Exploration

3.2.1 Mutual Information Based Variable Grouping

In the previous section, we demonstrated that given two variables, how specific information measures can be applied to detect salient isosurfaces. But, a multivariate system can consist of a large number of variables and selection of variables for further analysis plays an important role for future exploration stages. Since specific information measures are derived from mutual information, variables that have high mutual information, i.e., variables with high information overlap will be more suitable for further specific information based analysis.

From an information-theoretic point of view, each variable in a multivariate dataset carries a certain amount of information that is also shared by other variables, which can be characterized by the mutual information measure. An experiment with the Isabel Hurricane dataset reveals that there exist variables in the system which show strong correlation with some variables but not so much with others. In Figure 3.2, we show different instances of the relationship among the variables of the system. From the scatter plots shown in Figure 3.2(a) and 3.2(b), it is observed that a stronger relationship exists between the variables under study. A quick comparison of the scatter plots shown in Figure 3.2(c), 3.2(d), 3.2(e) and 3.2(f), on the other hand, reveals that the correlations between those variables are not as strong. This study of variable relationship allows us to perform a more systematic analysis of the variables using the specific information metrics.



(a) Cloud vs QIce.

(b) Pressure vs QRain.

(c) V-vel vs U-vel.

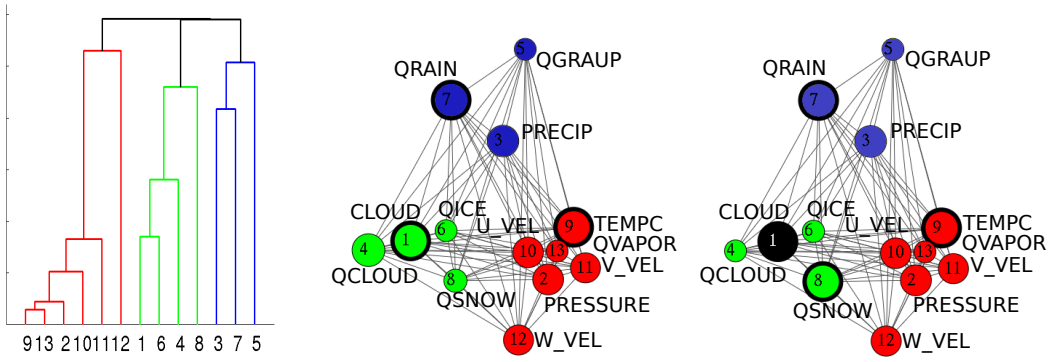
(d) Generated isosurfaces for the selected isovalue of the I_1I_2 map.

(e) QCloud vs QGraup.

(f) W-velocity vs QSnow.

Figure 3.2: Scatter plots between different variables showing different degrees of correlation..

For each pair of variables, we measure the distance between the variables as the inverse of the mutual information between them. Considering all the variables of the system, a graph $G(V,E)$ is constructed which delineates our system of variables. Each node $v \in V$ represents a variable, and each undirected edge $e \in E$ represents the mutual information between the two variables. A hierarchical clustering is applied to this graph to decompose it into different groups. In a bottom-up clustering approach, each node represents a leaf of the cluster tree and each of them starts out as a cluster. Then the new groups are formed via a greedy algorithm which merges the two clusters with the most similarity to move up the cluster tree one level. For n nodes, the general complexity of the algorithm is $O(n^3)$ and gives a locally optimal solution.



(a) The hierarchical clustering of the system of variables.

(b) The corresponding graph layout.

(c) The graph layout after variable selection.

Figure 3.3: Results from Isabel Hurricane dataset.

As shown in Figure 3.3(a), the dendrogram representation of the clustered graph is presented which reveals the hierarchy of the clusters. It is evident from the figure that there exist three major subgroups which form clusters according to the information overlap. In Figure 3.3(b), an alternative graph view of the system is provided. The layout of the graph is generated by a force-directed algorithm where the attractive force is given as the mutual information between the nodes and the repulsive force is the inverse of the mutual information. Here we see that the force-directed layout matches our hierarchical clustering results.

3.2.2 Information Based Variable Selection

For a collection of random variables $X_1, ..., X_n$, the total information content of the variables can be expressed by the calculation of joint entropy which is defined as

$$H(X_1, ..., X_n) = -\sum_{x_1 \in X_1} ... \sum_{x_n \in X_n} p(x_1, ..., x_n) \log p(x_1, ..., x_n). \tag{3.5}$$

As discussed in the previous section, in the multivariate datasets, the variables can be grouped based on their correlation, and the joint entropy can now be applied to measure the total uncertainty within each group using Equation 3.5 where the joint probability distribution of the variables is used. This allows us to compute the relative importance of the groups based on their uncertainty. The groups can be selected depending on their uncertainty and the individual variables inside the selected group now need to be analyzed.

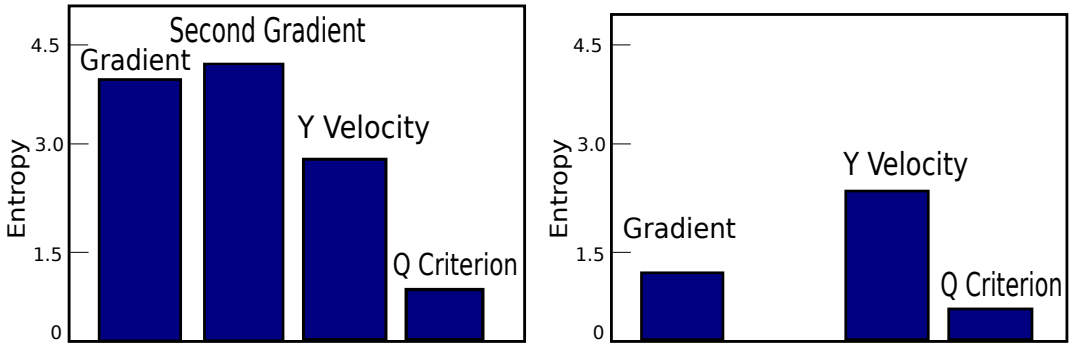
In information theory, given a group of variables, conditional entropy is used to quantify the information gain about the system when some of its variables are known. From n variables $X_1,...,X_n$, if m variables $X_{k1},...,X_{km}$ are known, then the amount of uncertainty left in the system is given by

$$H(X_1,...,X_n|X_{k1},...,X_{km}) = H(X_1,...,X_n) - H(X_{k1},...,X_{km}). \tag{3.6}$$

This provides a useful measure to identify variables inside subgroups that have a larger contribution towards the total uncertainty of a group. Also, this metric allows us to select some of the variables such that those variables represent the uncertainty of the whole subgroup by their information content.

An experiment with four fields of the Plume dataset is used as a motivating example of how the selection of one variable can reduce the uncertainty about other variables. Plume is a simulation of the thermal downflow plumes on the surface of the sun with $126 \times 126 \times 512$ grid points. Figure 3.4 shows that if we want to explore the higher uncertain variables first, then just by looking at their individual entropies, the order of selection would be Second Gradient, Gradient, Y Velocity and Q Criterion as shown in Figure 3.4(a). But, if Second Gradient field has been selected, it will now have impact on our knowledge about the other variables and as shown in Figure 3.4(b), the second variable to be chosen will be Y Velocity

as the uncertainty about Gradient field has been reduced due to the selection of Second Gradient field.



(a) Uncertainty remaining in the variables before selection.

(b) Uncertainty remaining in the variables after selection of Second Gradient.

Figure 3.4: Change in uncertainty of the variables due to the variable selection.

The objective of our variable selection methodology is to maximize the information gain about the subsystem after every variable selection. The selected variables also become the candidates for reference variables used to compute the I-metrics mentioned in section 3.1. It is likely that a variable of high information content, i.e., high entropy, will carry more information shared by other variables, and therefore, a good candidate to be used as a reference variable. In this sequential selection process, variables are selected and the residual information content of the remaining variables is updated to reflect the current state of the subsystem. Variables can be selected within a group and also across groups. While selecting variables across the groups, for each group, a variable can be identified which represents that subgroup and these representative variables can now be candidates for selection. For exploration within the group, the dynamic ordering of importance can be followed to guide the selection process.

3.3 PCP Based Uncertain Isovalue Identification

For the exploration in data domain, Parallel Coordinate Plots (PCPs) can be effectively used to visualize the system of variables. In PCP, the ordering of the variables is an important aspect for providing a useful visualization. Since the variables are clustered into groups based on the mutual information, we use this grouping information to control the display and to find the ordering for the PCP, as described below.

Because the groups represent variables with higher correlation, the variables can be shown group-wise in the PCP. Inside the group, we find an ordering of the variables such that the mutual information is highest between the neighboring axes of the PCP, so that user can more easily understand the relationships among the multiple variables. To order the variables into a sequence of PCP axes, our goal is to maximize the total amount of information presented in the plot. With our graph model, this ordering can be solved by finding a Hamiltonian path inside the subgraph that minimizes the sum of the edge weights. As the edge weights are the inverse of the mutual information, the minimum weight Hamiltonian path maximizes the total information presented along the sequence of PCP axes. Although finding an optimum Hamiltonian path in a graph is an NP-complete problem, for relatively small graphs, the brute force method works well. For larger graphs, an approximated solution minimizing the inter-cluster crossings yields sufficiently good results [146].

To supply data to the PCP interface, the users can choose to explore the volumetric multivariate dataset in slices, multiple slices or the whole volume. However for large volume data, as the number of points grows larger, the PCP can get cluttered. Brushing is a popular technique in visualization to interactively select the regions of interest. We allow brushing of points according to the scalar values and also according to the specific information values. Since the specific information metric I_2 represents the uncertainty or predictability

factor and I_1 describes the "surprise" of the scalar value, to show the scalar values which correspond to higher uncertainty in the other variable, a derived metric I_{uncert} is formulated such that

$$I_{uncert} = I_1 / I_2. \tag{3.7}$$

I_{uncert} identifies the higher I_1 and lower I_2 values. Similarly, for the scalar values which correspond to lower uncertainty in the other variable, another derived metric I_{cert} is generated which is given as

$$I_{cert} = I_1 * I_2. \tag{3.8}$$

I_{cert} identifies the higher I_1 and higher I_2 values.

3.4 Framework Overview

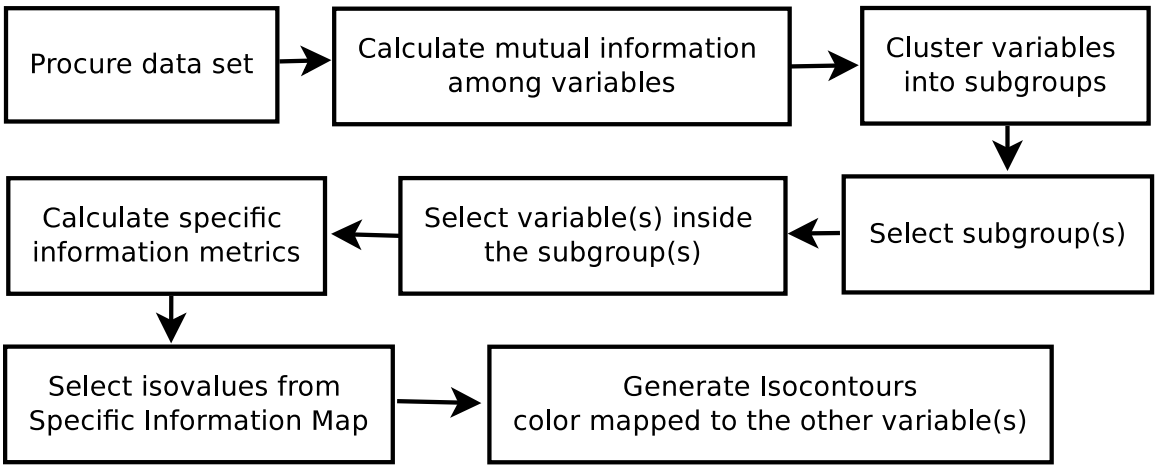


Figure 3.5: A schematic representation of the workflow.

A schematic view of our proposed system is provided in Figure 3.5. Here, we demonstrate that by using the concept of specific information measures, one can quantify the

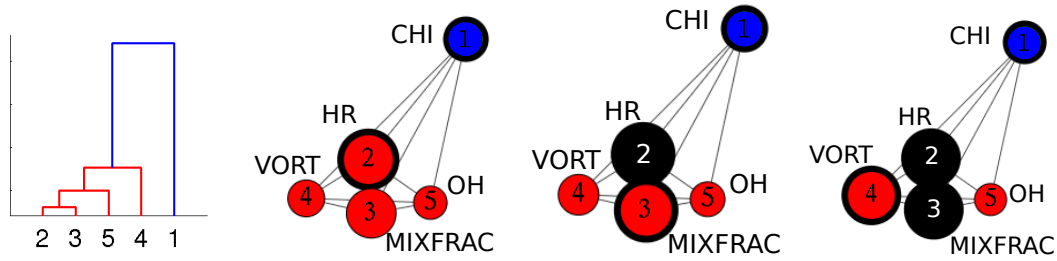
informativeness of a scalar value in one variable with respect to the uncertainty of the other variables. The specific information is based on the decomposition of the standard mutual information. To allow for more effective analysis of correlation in the dataset, the variables under study should have a good information overlap. To achieve this, the variables are subdivided into smaller groups based on their mutual information. The subgroups can be selected based on their total information content which is measured by the joint entropy of the subsystem. Inside the subgroups, the variables that contain the most information about the subgroup are identified using conditional entropy. After selecting the variables, the specific information metrics are used to identify different scalar values with varying amounts of information. We present an information-aware multivariate data exploration framework with novel features such as quantitative analysis of information overlap among the variables, information-driven grouping and selection of variables for more systematic correlation study, and flexible user interface for easy selection and browsing of salient data.

3.5 Case Studies

In this section, we discuss our uncertainty guided isovalue selection for three multivariate datasets. The experiments were conducted on a Linux machine with an Intel core i7-2600 CPU, 16 GB of RAM and an NVIDIA Geforce GTX 560 GPU with 2GB texture memory. For the calculation of information-theoretic measures 256 histogram bins were used. The force directed graph layout was generated by the Boost Graph Library [124].

3.5.1 Combustion Dataset

This dataset is a turbulent combustion simulation data which is a time-varying volume dataset having five scalar variables: Mixture Fraction (MIXFRAC), Vorticity (VORT), Mass Fraction of Hydroxyl (OH) radical, Heat Release Rate (HR) and Scalar Dissipation



(a) Hierarchical clustering of the variables.

(b) Graph layout of the variables.

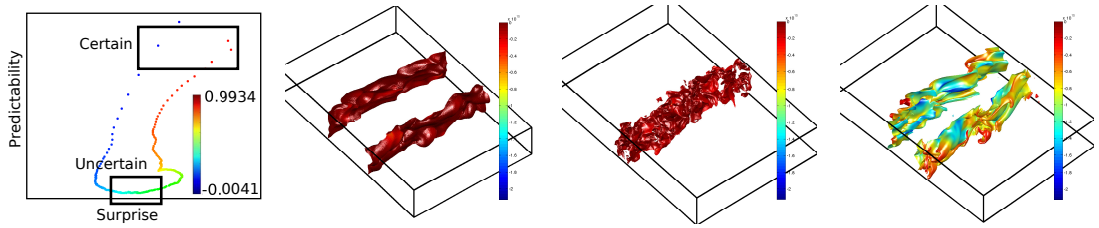
(c) State of the graph after first variable selection.

(d) State of the graph after second variable selection.

Figure 3.6: Combustion dataset with 5 variables: hierarchical clustering based on mutual information and a force-directed graph layout.

Rate (CHI) in turbulent flames. The dataset is made available by Dr. Jacqueline Chen at Sandia Laboratories through US Department of Energy's SciDAC Institute for Ultrascale Visualization. Each time step of this dataset contains $480 \times 720 \times 120$ grid points. The mixture fraction denotes the proportion of fuel and oxidizer mass and this value generally provides the location of the flame where the chemical reaction rate exceeds the turbulent mixing rate. But there can be regions where the mixing rate dominates the chemical reaction rate and the flame is partially extinguished or weakly burning. To have a detailed understanding of the combustion phenomenon, only analyzing the mixture fraction may not be enough and multivariate analysis is needed for this complex process. Time step 41 from the dataset was selected for our experimental purposes.

Using our framework, the all pair mutual information is calculated and then the graph layout is generated for user interaction as shown in Figure 3.6. After mutual information based grouping, Mixture Fraction, Heat Release Rate, Vorticity and OH mass fraction are placed in the same subgroup and this subgroup has higher entropy compared to the subgroup consisting of Scalar Dissipation Rate as reflected by the color of the subgroups in

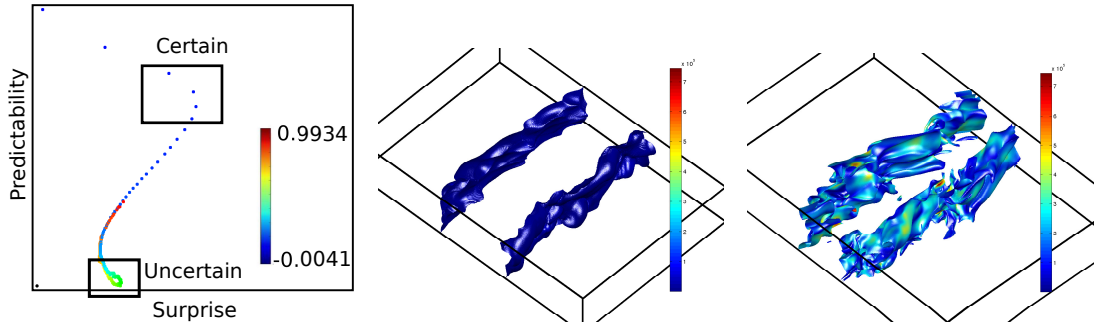


(a) I_1I_2 map of Mixture Fraction computed against Heat Release Rate.

(b) Mixture Fraction isosurface for isovalue 0.01 colored by the Heat Release Rate showing low uncertainty.

(c) Mixture Fraction isosurface for isovalue 0.95 colored by the Heat Release Rate showing low uncertainty.

(d) Mixture Fraction isosurface for isovalue 0.40 colored by the Heat Release Rate showing higher uncertainty.



(e) I_1I_2 map of Mixture Fraction computed against Vorticity.

(f) Mixture Fraction isosurface colored by the Vorticity. At low Mixture Fraction value, 0.01, the uncertainty about Vorticity is low.

(g) Mixture Fraction isosurface colored by the Vorticity. In this selection, at Mixture Fraction value 0.5, the uncertainty about Vorticity is much higher.

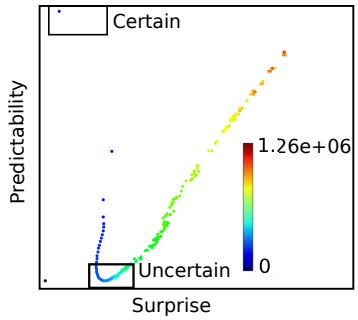
Figure 3.7: Different isosurfaces showing different amount of uncertainty for other variables for Combustion dataset.

the graph layout. Then, we proceed to the exploration of the individual variables. In the process of variable selection within the subgroup, the first two variables suggested by our framework are Heat Release Rate and Mixture Fraction. The corresponding I_1I_2 map of the reference variable Mixture Fraction is shown in Figure 3.7(a). Selecting the points which correspond to high I_1 and high I_2, it is observed that isosurfaces of very low and very high Mixture Fraction values all have very high Heat Release Rate, as shown with the two selected isosurfaces in Figure 3.7(b) and Figure 3.7(c). But for the Mixture Fraction values

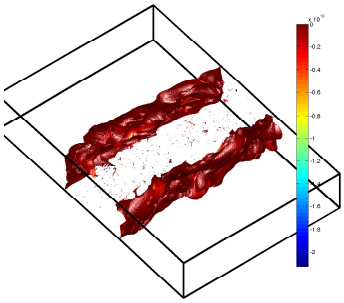
between 0.3 to 0.55, the Heat Release Rate values are varying widely. As mentioned in [2], the stoichiometric mixture fraction for this mixture is 0.42 which corresponds to the flame. From our results, it is observed that the Heat Release Rate is not stable on or around the flame which suggests that some complex interaction around the flame is happening. To get more insight, the next variable to be analyzed is Vorticity or the flow turbulence, as suggested by our graph layout. The Figures 3.7(e),3.7(f) and 3.7(g) present the results of isosurfaces drawn on the reference variable Mixture Fraction color mapped with Vorticity. From the results, it is to be understood that near the flame, the turbulence is higher, which is causing higher uncertainty in the region. Similar deductions can be made from the Figures 3.8(a), 3.8(b), and 3.8(c) where we show the results of Vorticity and Heat Release Rate. The more certain Heat Release Rate values occur where the reference variable Vorticity is much lower. As the Vorticity or the turbulence increases, the Heat Release Rate becomes more uncertain. Finally, we present the results of the analysis of Mixture Fraction and Mass fraction of OH radical in the Figures 3.8(d), 3.8(e) and 3.8(f) with Mixture Fraction as the reference variable. From the resulting isosurfaces, it is apparent that the OH mass fraction is not constant around the stoichiometric mixture fraction as noted in [2].

3.5.2 Hurricane Isabel Dataset

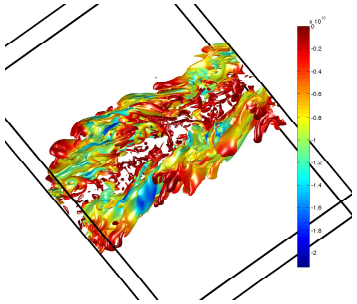
Next, we show the exploration results from the Hurricane Isabel dataset. Hurricane Isabel data was produced by the Weather Research and Forecast (WRF) model, courtesy of NCAR and the U.S. National Science Foundation (NSF). This dataset consists of thirteen variables and the resolution of the dataset is $500 \times 500 \times 100$ for a single time step. We have selected time step 20 for our experiments.



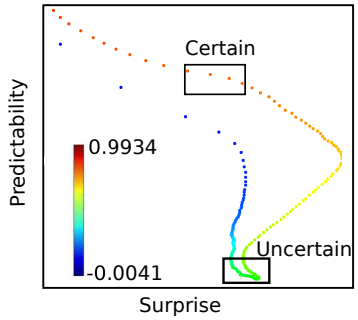
(a) I_1I_2 map of Vorticity computed against Heat Release Rate.



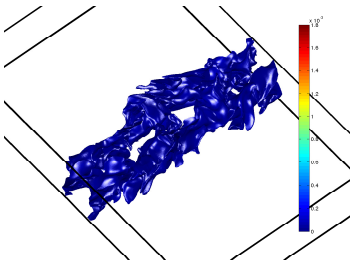
(b) Vorticity isosurface colored by the Heat Release Rate. At very low Vorticity, the uncertainty about HR is low.



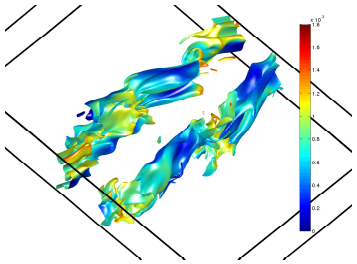
(c) Vorticity isosurface for isovalue 148750 colored by HR. At higher Vorticity value, the uncertainty about HR is higher.



(d) I_1I_2 map of Mixture Fraction computed against Oxidizer OH mass fraction.



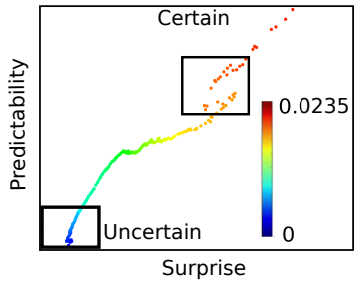
(e) Mixture Fraction isosurface colored by the Oxidizer OH mass fraction. At Mixture Fraction 0.85, the uncertainty about OH mass fraction is low.



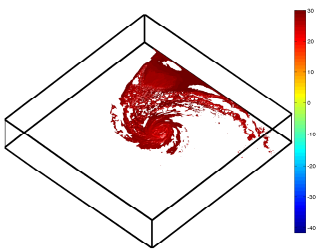
(f) Mixture Fraction isosurface colored by the Oxidizer OH mass fraction. At Mixture Fraction value 0.47, the uncertainty about OH mass fraction is high.

Figure 3.8: Different isosurfaces showing different amount of uncertainty for other variables for Combustion dataset.

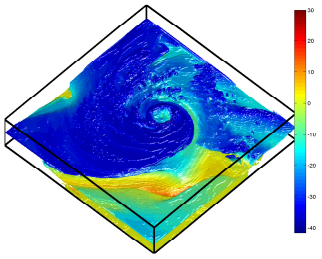
For this dataset, the initial force-directed graph layout was presented in Figure 3.3(b). When clustering the graph into three groups, the highest entropy group consists of Pressure, TC, U Vel, V Vel, W Vel, and QVAPOR as reflected by the color of this group shown in the graph layout. From this group, choosing the variables Temperature and QVAPOR, with QVAPOR as the reference variable, we constructed the I_1I_2 map as shown in Figure 3.9(a). From this map, the QVAPOR values are selected which correspond to less variability in



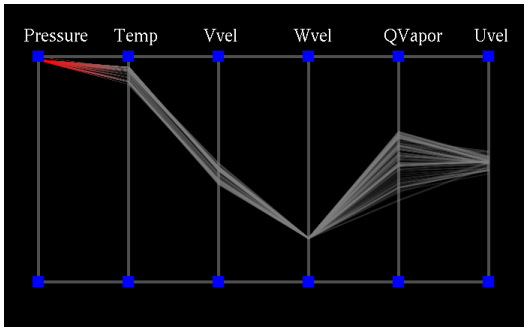
(a) I_1I_2 map of QVAPOR computed against Temperature.



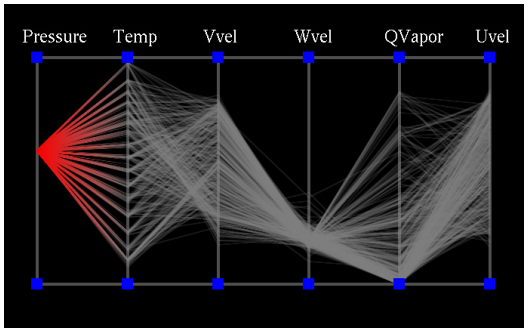
(b) QVAPOR isosurface colored by the Temperature. At QVAPOR value 0.0188, the uncertainty about Temperature is low.



(c) QVAPOR isosurface colored by the Temperature. At QVAPOR value 0.0004, the uncertainty about Temperature is much higher.



(d) Selection of certain scalars from PCP using I_{cert}.



(e) Selection of uncertain scalars from PCP using I_{uncert}.

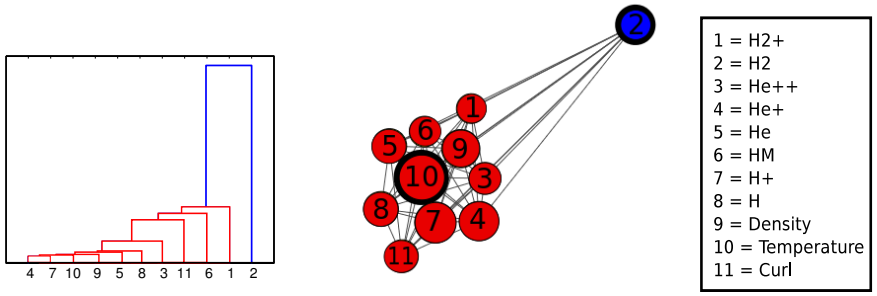
Figure 3.9: Different isosurfaces showing different amount of uncertainty for other variables for Isabel dataset.

Temperature as shown in 3.9(b). On this isosurface, which is generated by QVAPOR value 0.0188, nearly the same and relatively high Temperature values are observed. Whereas, we can also identify isosurfaces as in the Figure 3.9(c), where the variability in the Temperature value on the surface is much higher for QVAPOR value close to 0.0004. Figure 3.3 shows the result of selecting Pressure and U velocity for analysis, where Pressure is the reference variable. The I_1I_2 map is presented in 3.1(a) which is used to select certain and uncertain isosurfaces on Pressure corresponding to U velocity. Figure 3.1(b) shows examples of two such isosurfaces. In this figure, the Pressure isosurface for isovalue 130 has much less

variation in U Velocity whereas, as we move closer to the Hurricane eye region where the Pressure goes down, there is much more variation in U Velocity as represented by the isosurface of Pressure value around -700.

For a downsampled version of Isabel data, we use PCP to highlight informative scalars for Pressure and Temperature in Figures 3.9(d)-3.9(e) where Pressure is used as the reference variable for specific information calculation. In this figure, we only show the variables of the selected group for demonstration purposes. We use I_{cert} and I_{uncert} metrics to choose the scalars from PCP. Figure 3.9(d) shows the selection of scalar values using I_{cert} where a small range of scalars of Pressure maps to a small range of scalars in Temperature. Conversely, Figure 3.9(e) shows the selection of scalars using I_{uncert} where a small range of scalar values of Pressure maps to a much larger range of scalar values on the Temperature axis.

3.5.3 Ionization Front Instability Simulation Dataset

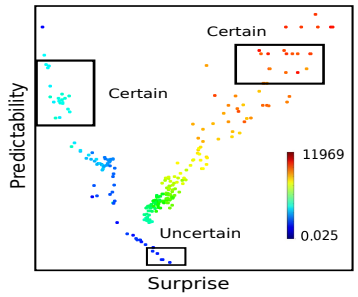


(a) Dendrogram representation of the hierarchical clustering of the variables.

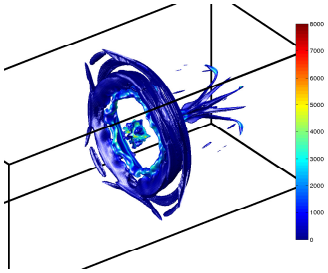
(b) Force-directed layout for the graph view of the variables.

(c) Variable names.

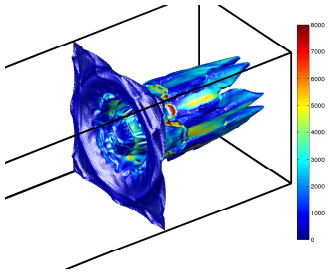
Figure 3.10: Different isosurfaces showing different amount of uncertainty for other variables for Isabel dataset.



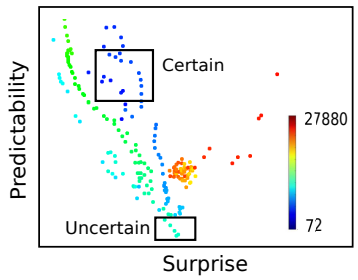
(a) I_1I_2 map of H computed against Curl.



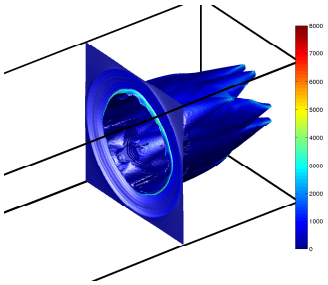
(b) H isosurface colored by Curl magnitude. At low species number density of H, e.g. 235, the uncertainty about Curl is low.



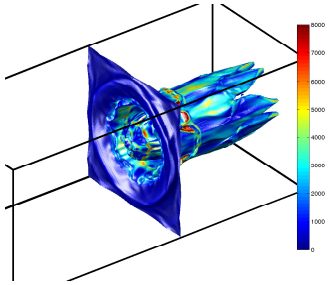
(c) H isosurface colored by Curl magnitude. At intermediate species number density of H, e.g. 3285, the uncertainty about Curl is much higher.



(d) I_1I_2 map of Temperature computed against Curl magnitude.



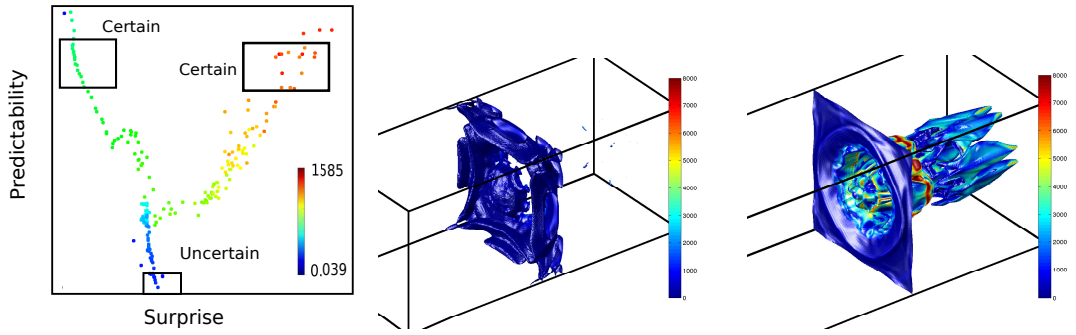
(e) Temperature isosurface colored by Curl magnitude. At low Temperature, e.g. 1150, the uncertainty about Curl is low.



(f) Temperature isosurface colored by Curl magnitude. At higher Temperature, e.g. 8800, the uncertainty about Curl is much higher.

Figure 3.11: Results of Ion Front dataset.

In our next case study, we use the dataset presented in the IEEE 2008 Visualization Design Contest [142]. In this dataset, researchers have intended to explore the relationships of the ionization front instabilities with the formation of the first stars of the universe. This dataset contains the relative abundancies of eight chemical species, temperature, density and the velocity field. The dataset size is $600 \times 248 \times 248$ and we have selected time step 99 for exploration purposes. One of the tasks of the contest was to investigate the reasons of turbulence. We used the curl magnitude computed from the velocity field as the turbulence



(a) I_1I_2 map of H+ computed against Curl magnitude.

(b) H+ isosurface colored by Curl magnitude. At higher species number density of H+, e.g. 620, the uncertainty about Curl is low.

(c) H+ isosurface colored by Curl magnitude. At low species number density of H+, e.g. 62, the uncertainty about Curl is much higher.

Figure 3.12: Results of Ion Front dataset.

measure and calculated the individual species number densities of the chemicals to show our exploration results.

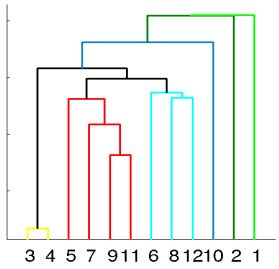
The initial clustering result and layout of the force-directed graph is presented in Figure 3.10(a) and Figure 3.10(b). To see the relationship between Curl and H, the corresponding I_1I_2 map was generated as shown in Figure 3.11(a) using H as the reference variable. In this case, we find that the low and high species number densities of H correspond to more certain regions in Curl. Figure 3.11(b) shows an isosurface at a low value of H. The uncertain regions occur in the intermediate values of H species density where there is more variability in Curl values, shown in Figure 3.11(c). From our framework, if the node representing Curl is selected, then Temperature becomes the next candidate for selection. The corresponding results are shown in Figures 3.11(d), 3.11(e), and 3.11(f). In this case, lower temperature values correspond to more certain Curl values and with the increase in Temperature, the Curl values become more uncertain as shown by the two isosurfaces. If Curl and Temperature nodes are selected, then the next candidate node suggested by our system

is H+. The Figures 3.12(a), 3.12(b), and 3.12(c) represent results where lower values of H+ have more uncertain Curl values, whereas the higher H+ values correspond to more certain Curl values.

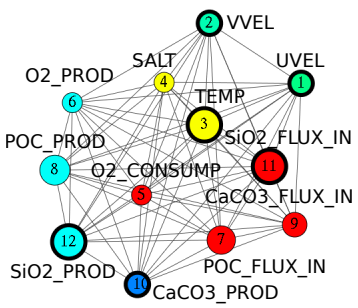
3.6 Domain Expert's Feedback

We demonstrated our system to the scientists of the Los Alamos National Laboratory to assess the effectiveness of the system for real world applications. At the beginning of the demonstration, high level explanation about the system workflow was presented to the scientists. Then we went over the details of each component of the system, providing some pre-generated results and images for an understanding of how the system helps in the exploration of the multivariate datasets. As the domain experts gained familiarity with the system, we presented the Eastern Seaboard and Gulf of Mexico region from the ocean component (LANL's POP model) of a fully coupled climate simulation using the DOE/NSF Community Climate System Model (CCSM4) with a grid resolution of approximately 1 degree and resolution of $320 \times 384 \times 60$. Embedded within the ocean is a model of marine ecodynamics that includes representations of plankton, nutrients, and other forms of organic and inorganic matter. This demonstration included several variables that described the production ("PROD") and sinking ("FLUX-IN") of Particulate Organic Carbon (POC), Silicate (SiO2), and Calcium Carbonate (CaCO3), as well as Oxygen (O2) production and consumption (CONSUMP), Temperature (TEMP), Salinity (SALT), and the horizontal velocity components (UVEL, VVEL).

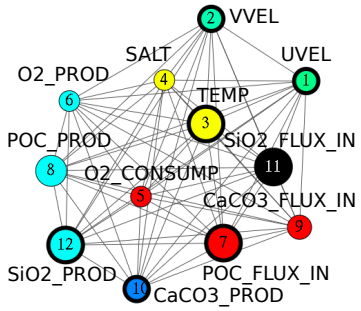
In the exploration process, the users initially followed the suggestions made by the system and later on, they explored some of the variables by themselves. At the beginning, the users interactively selected the number of clusters of the hierarchical cluster tree to



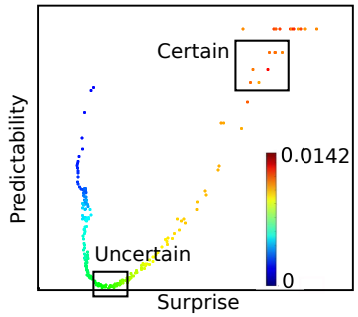
(a) Hierarchical cluster tree clustered into 6 groups.



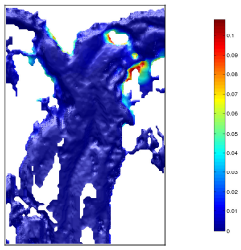
(b) The graph layout.



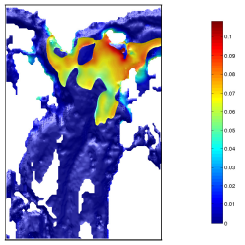
(c) State of the graph after a variable selection.



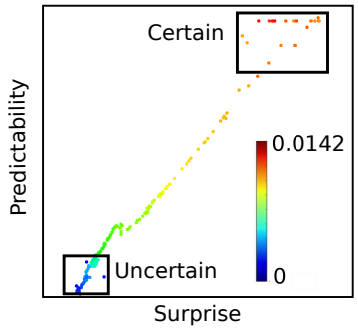
(d) I_1I_2 map of SiO2-FLUX-IN against POC-FLUX-IN.



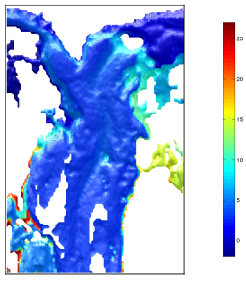
(e) Isosurface of SiO2-FLUX-IN for isovalue 0.0124 color mapped by POC-FLUX-IN.



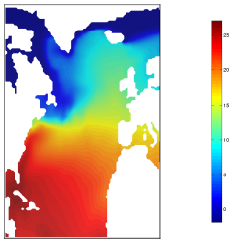
(f) Isosurface of SiO2-FLUX-IN for isovalue 0.0075 color mapped by POC-FLUX-IN.



(g) I_1I_2 map of SiO2-FLUX-IN against TEMP.



(h) Isosurface of SiO2-FLUX-IN for isovalue 0.0128 color mapped by TEMP.



(i) Isosurface of SiO2-FLUX-IN for isovalue near 0 color mapped by TEMP.

Figure 3.13: Application of our system on POP dataset for domain expert's feedback.

begin the exploration. Figure 3.13(a) shows the hierarchical cluster of the given set of variables where the number of clusters was 6. This grouping based on information overlap

provides insight into the data. For example, the grouping of SiO2-FLUX-IN, CaCO3-FLUX-IN, POC-FLUX-IN and O2-CONSUMP points to the fact that in the ocean, vertical fluxes of phytoplanktonic hard parts are often correlated and the particulate organics exert demand on dissolved oxygen. Then the users could learn about the information content of the subgroups by observing the node colors of the graph as shown in Figure 3.13(b). For exploration within a subgroup, the users initially chose the variables SiO2-FLUX-IN and POC-FLUX-IN as suggested by the graph in Figure 3.13(b) and Figure 3.13(c). The corresponding I_1I_2 map was generated as Figure 3.13(d) with SiO2-FLUX-IN as the reference variable. The map shows that the low and the high values of the reference variable have higher predictability but the higher values have more surprise. Then from this map, isovalues were selected according to their Predictability and Surprise for visualization of the isosurfaces as shown in Figure 3.13(e) and Figure 3.13(f) to show the variability of the POC-FLUX-IN. For exploration across the subgroups, the two representative variables SiO2-FLUX-IN and TEMP were selected and the corresponding I_1I_2 map was generated with SiO2-FLUX-IN as the reference variable which is shown in Figure 3.13(g). From this map, it is evident that larger values of SiO2-FLUX-IN have higher predictability and surprise compared to its smaller values. This points to the fact that for this dataset, the silica concentration is higher at the bottom level of the ocean where the cooler and heavier water is present which gives lower variability in temperature. Conversely, at the surface level, the silica concentration is lower and temperature variability is higher. This can be visually analyzed by generating isosurfaces of SiO2-FLUX-IN and color-mapping it with TEMP. Figure 3.13(h) shows much higher predictability where the selected isovalue is 0.0128 and Figure 3.13(i) shows much more uncertainty about TEMP when a SiO2-FLUX-IN value close to 0 is selected which gives the surface of the ocean. Apart from the variables

suggested by the system, the scientists also explored some other variables that they were interested in and they verified the outcomes of our system with their knowledge base.

After the entire process was completed, we asked for general and specific feedback about our system. From the feedback provided by the domain scientists, it is apparent that they are looking for new tools and are eager to try out systems similar to what we have proposed in this chapter. Given a multivariate dataset, exploration of the relationships among the variables is of prime interest to them. The scientists noted that the initial dendrogram representation tied to the graph layout provided them with new ways to identify classic expected relationships among the marine systems variables but additionally the visualizations raised new kinds of research issues which they felt was very useful. If the datasets do not show the relationship pattern as expected by the scientists, they can flag this timestep for more analysis as to whether this anomaly is due to some error in the simulation or indeed something unexpected has happened. The I_1I_2 map provides information about the predictability and surprise of the reference variable with respect to the other variable. The final visualization of the color-mapped isosurfaces can also effectively convey the variability information of the other variables when compared with the reference variable. In the scientists' opinion, this system provides information about the variables from a new perspective with an interactive interface which they think is useful for multivariate exploration. The scientists concluded that besides existing tools like Matlab and Ferret, they would like to use our system for multivariate analysis in the future.

The scientists also provided suggestions to improve the system. They want to see our system integrated with the existing tools like scatter plot matrices so that it conveys more information. They also would like to see this exploration system to be extended in temporal domain for increased effectiveness. Specific to their domain needs, they would also

like to explore in isopycnals (the constant density slices) that can reveal the mixing in the flow. In the graph layout, they suggested to add an option to remove the edges as they are liable to produce visual clutter when the number of variables is high. It was also apparent from the feedback that sometimes the domain scientists had some pre-selected variables in mind, like Salt, Temperature etc., which they were interested in exploring regardless of their information content. While sometimes the variables suggested by our initial graph structure did not match their primary interest, they agreed that for a large number of unfamiliar variables in the system, the knowledge of the relative information content is useful in making a choice.

3.7 Performance

Table 3.1: Running Time for Different Components of the Framework

	Map Creation (Sec.)	All Pair M.I. (Sec.)	Joint Entropy (Sec.)	I_1I_2 Map Generation (Sec.)
Combustion	25.9	86.5	12.3	10.6
Isabel	38.8	1071.6	53.9	6.3
Ion Front	43.7	92.5	11.4	8.5

In this section, we discuss the performance of different components in our proposed framework as shown in Table 3.1. To create the joint distributions among multiple variables, instead of creating a large multi-dimensional array which will be very expensive in terms of the required storage space, we took advantage of the sparseness property in the array and used the Map data structure provided by C++ library to perform the Joint Entropy and all pair Mutual Information (M.I.) calculations. Without including the disk I/O time, the map creation time shown in the first column is dependent on the total number of data

points for all the variables of the system. In all pair M.I. calculation from the generated data map, it is observed that the run time varies depending on the distribution of the data values on the map. This all pair M.I. calculation is used to generate the force-directed layout of the graph. The run time of the hierarchical clustering and force-directed layout calculation is not listed here as it is in the order of milliseconds. Next, we presented the Joint Entropy calculation time within a group of variables which is required to figure out the relative importance of the variables. This is dependent on the group size and the distribution of the values in the data map. In the third column of the table, we have listed the longest running time which was measured by the time taken to update the importance of the variables belonging to the largest group. In our experimental settings, the largest group sizes are 4, 6, and 10 for Combustion, Isabel, and Ion Front dataset, respectively. Finally, the I_1I_2 map is generated from two selected variables and as shown in the fourth column, according to our current implementation, the calculation time is linear to the number of points in the datasets for a fixed number of bins. Generation of the maps, all pair mutual information and the joint entropy calculations can be done in a preprocessing stage so that our framework can run interactively.

3.8 Conclusions

In this chapter, we discussed our uncertainty based isocontour selection via an information-theoretic multivariate exploration framework. In our framework, we employed specific information based measures to compute the saliency of the isocontours that also provide uncertainty information regarding the selected variables. Given a multivariate dataset, we applied mutual information to generate an initial grouping of the variables such that the variables with high information overlap can be placed in the same group. The importance

of the variables within the subgroups is identified by the calculation of conditional entropy. The variables are presented in the form of a force-directed graph layout to the users for interactive selection of variables. The selected variables are used to compute the specific information to identify the scalar values of one variable which are informative about the other variables. For the exploration in the data domain, the visualization is performed using Parallel Coordinate Plots. For exploration in the spatial domain, isosurfaces are generated which are color mapped to other variables to show the degree of uncertainty about that variable.

Chapter 4: Visualization of Time-Varying Weather Ensembles Across Multiple Resolutions

From the analysis of multivariate relationships, we now focus on the relationships of parameters with the output of the ensemble datasets in this chapter. Ensemble climate and weather simulation models are of prime importance to the scientists for analyzing and predicting future climate changes. For a reliable decision-making from these models, a robust prediction is needed for most likely scenarios (mean behavior) as well as low-probability but potentially hazardous events (outliers). To provide such robust predictions, full knowledge of the output, i.e. the full probability density function of the output is needed. Due to the lack of this ground truth, the output density function is generally approximated by running a set of ensemble simulations. A collection of techniques has been employed to generate ensemble simulations of climate model output such as using different simulation models, sampling the input parameter space, changing the initial condition of the simulations etc. Among these, perturbation of input parameters is most popular and well-accepted [149] for generating ensembles of rainfall and other climate model outputs.

Despite the advancements in computation power and scientific capabilities, the true nature of how the different input parameters interact among themselves and with respect to the output remains unknown. Due to the presence of non-linear interactions [140] among the physical processes, uncertainty quantification (UQ) of the model inputs and

outputs is of prime importance. In climate modeling, physics parameterizations generally use conceptual or empirical relationships to approximate the impact of sub-grid processes on the resolved-scale dynamics and thermodynamics. Consequently, the parameterization schemes may contain some parameters that have no direct physical equivalence in nature and are subject to a certain degree of uncertainties. It remains unclear that such parametric uncertainty, parameter tuning, and calibration may lead to different results in different regions and climate regimes. Thus a thorough analysis of the input parameters is required for good quality weather and climate prediction [16, 55, 134].

In this chapter, we use the Weather Research and Forecasting (WRF) regional climate model to examine the sensitivity and uncertainty on five convective parameters in the Kain-Fritsch (KF) convection parameterization scheme (CPS). The motivation of this study is to 1) investigate the spatial and temporal variation of parametric sensitivity and parameter estimation for the Kain-Fritsch convective scheme based on regional data under three different resolutions in the WRF model; 2) evaluate the trade-off between efficiency and accuracy among the ensemble of WRF model simulations under three different resolutions; 3) visualize the parametric sensitivity and uncertainty for the Kain-Fritsch convective scheme based on regional data under three different resolutions. To address these needs of the domain experts, we conducted a series of simulations to model the precipitation over the Southern Great Plains (SGP) area with the grid spacing of 12, 25, and 50 km (different spatial resolutions) and using two radiation schemes (RRTMG versus CAM) (different physical sub-models). The simulations produce a time-varying ensemble dataset across three resolutions along with their input parameterizations and with such high dimensional dataset, visualization and analysis become non-trivial.

In this chapter, we propose a workflow for exploration of multi-resolution time-varying ensemble datasets that addresses the needs of the domain scientists. We attempt to answer two very specific questions posed by the domain experts: (1) How are the parametric sensitivity, parameter tuning and calibration processes sensitive to grid spacing, grid locations, and temporal domain? (2) Are the optimal results transferable across spatial resolutions and how does the model accuracy change across resolutions?

For exploring the sensitivity of the input parameters against the output precipitation, a sensitivity measure named *delta* [15] is employed. Among the other existing sensitivity measures, this method is chosen since it is global, moment independent and popular in the operations research (OR) community. We apply this sensitivity measure across spatial locations and across time-domain to explore the intra and inter-resolution sensitivity trends. With sensitivity values distributed across spatial locations, a local clustering exposes the spatial trend in the sensitivity values. This clustering reveals the change of sensitivity with the model output and provides insight into which parameters are to be tuned finer for faster convergence of the simulations. To understand the temporal trend of the sensitivity values across resolutions, we apply a dimensionality reduction approach to create a two-dimensional layout along with a line chart view and allow the users to interactively explore the entire data domain by connecting it with the spatial view. To explore the efficiency and accuracy of the three resolutions, we take up a non-parametric kernel density estimation approach to analyze the ensemble precipitation outputs. Given the observed precipitation data, we employ Bayesian likelihood analysis to identify which resolution has better prediction accuracy at different spatial locations. After computing these local best predictors, the time-varying data is aggregated into one final image using custom color-maps that guides the future simulations towards an adaptive grid implementation. For the detailed

exploration of accuracy trends of the multi-resolution dataset, we design an interface with brushing and linking of the temporal and spatial domains. Our designed exploration tool uses domain experts' inputs and helps them understand the multi-resolution temporal ensemble dataset. We work closely with a domain expert at each stage of the pipeline to verify and answer his domain specific needs. The domain expert's feedback shows that our framework is effective in providing detailed insight into such a high-dimensional dataset.

4.1 Background and Overview

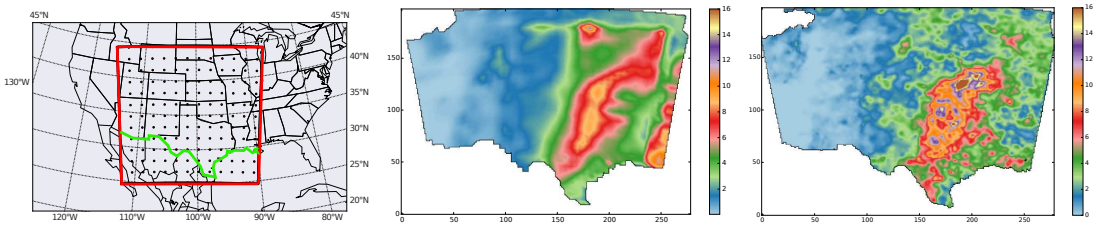
4.1.1 Motivation

Along with the accuracy of the model output, the input parameter sensitivity study has historically been of prime importance for weather ensembles, and it continues to be an active research area [54, 59, 148, 154]. Weather and climate simulation models often include multiple input parameters ($\approx$ 15 to 30) and there generally exists a non-linear relationship among the inputs and outputs in these models. For such complex models, a lot of computational power and resources are expended in high-dimensional input parameter space sampling for generating good quality ensemble outputs. Consequently, the climate modelers employ parameter sensitivity analysis to rank parameters [54, 148, 154] according to their influence. This enables the scientists to calibrate the model by tuning the most influential parameters of the system or perform experiments to better characterize those parameters. When the simulations are used to generate time-varying data at different spatial locations, this sensitivity information of the parameters can change over the spatial-temporal domain. Thus, a thorough exploration of how the sensitivity values of the model parameters change as a function of spatial location and over the temporal domain is of great importance to the domain experts.

Exploration of accuracy and parameter sensitivity becomes harder when the ensemble simulation output is generated for varying grid resolutions. Domain experts investigate multi-resolution datasets from simulation models since datasets of varying resolutions can capture different scales of physical phenomena. As different resolutions have different computational costs, domain experts need to understand the accuracy vs cost trade-offs for running future simulations. In climate simulations, as the model performance is sensitive to other physical parameterizations which also depend on model resolution, it is not guaranteed that higher resolution models always perform better than lower resolution models. For instance, it is observed that in some ensembles, precipitation results using 50km spacing grid can be more accurate than that using higher resolution 12km spacing grid. Hence it is critical to develop an effective data visualization tool to analyze simulation models for identifying the advantages and limitations of generating data at varying resolutions. The existing data analytics software, such as Paraview, VisIt, Tecplot, etc. do not integrate the multi-resolution aspect of the temporal ensembles for exploration of accuracy and sensitivity across spatial location and time-domain. Our proposed solution keeps the domain expert in the loop for a detailed and effective visualization-guided exploration of the entire dataset.

4.1.2 Data Description

WRF regional climate model is widely accepted mesoscale numerical weather prediction system and this model uses Kain-Fritsch (KF) convection parameterization scheme (CPS) to simulate convection. A set of five parameters is used in the simulations to generate the precipitation output. In particular, the five uncertain convective parameters (as input parameters to the ensemble WRF model simulations) are: Pd, coefficient related to



(a) (b) (c)

Figure 4.1: Visualization of the ensemble dataset and the ground truth. (a) Southern Great Plains (SGP) regions for the ensemble data (marked in red). The green contour marks the United States boundary. Observed data is only available for the north of the green contour inside the red box. (b) Precipitation output at 50km resolution averaged over all the ensemble runs through the month of June, 2007. (c) Monthly averaged observed rainfall for the month of June, 2007.

downdraft mass flux rate, Pe, coefficient related to entrainment mass flux rate, Pt, maximum turbulent kinetic energy, Ph, starting height of downdraft above updraft source layer, and Pc, average consumption time of convective available potential energy. The range of values and the default values used for each of the parameters are listed in Table 4.1. The model parameters were calibrated using simulated stochastic approximation annealing (SSAA) calibration algorithm. The SSAA algorithm used for global calibration is based on the Very Fast Simulated Annealing (VFSA) algorithm [84]. Using this approach, the sampling range of each parameter narrows at every stage of the convergence process. We also calibrate the CPS by constraining simulated precipitation with the observations.

The observation dataset consists of observed precipitation in June 2007 over the Southern Great Plains (SGP) area (latitude $\times$ longitude) $[25.0, 44.0] \times [-112.0, -90.0]$. The SGP coverage area is shown in Figure 4.1(a) and the monthly averaged observed precipitation is depicted in Figure 4.1(c). The computer simulation dataset consists of 150 ensemble simulations at different sets of 5 uncertain convective parameter values selected via Latin

Table 4.1: Description and range of the input parameters for the WRF model.

Parameter	Description	Range	Default
Pe	Coefficient related to entrainment mass flux rate	[-1,1]	0
Pd	Coefficient related to downdraft mass flux rate	[-1,1]	0
Pt	Maximum turbulent kinetic energy	[3,12]	5
Ph	Starting height of downdraft above updraft source layer	[50,350]	150
Pc	Average consumption time of convective available potential energy	[900,2700]	2700

Hypercube Sampling [86] over the SGP area with the grid spacing of 12, 25, and 50 km. This generates 150 ensemble simulated output precipitation at three different resolutions over the 30 days. The regions outside United States of America and ocean regions are marked with a large positive value in the ensemble dataset and are discarded from the analysis. Monthly average of the mean ensemble output for 50km resolution is shown in Figure 4.1(b).

4.1.3 Problem Statement

In the context of the WRF climate model, domain experts have a sequence of domain specific questions that we have answered in this chapter. The specific questions that we address here include the following:

1. Given the set of input parameters and the ensemble output, which parameters are most important in deciding the output precipitation?

2. How the sensitivity of the individual parameters change as a function of spatial location, time step and across different resolutions?

3. Do the sensitivity values of the different input parameters remain the same over different spatial locations and time steps for different resolutions? Or, how the relative importance of the parameters change?

4. How the different data resolutions are related to each other in terms of their input parameter sensitivities? Do the sensitivities change when the ensemble output is generated at varying resolutions?

5. In the presence of the ground truth, how does the prediction accuracy of the ensemble runs change for different resolutions? Does the higher resolution always have higher accuracy?

6. How to effectively explore the regions with varying prediction accuracy of a multi-resolution temporal ensemble dataset? How to quickly identify the regions with high prediction error for different resolutions?

4.1.4 Visualization System Overview

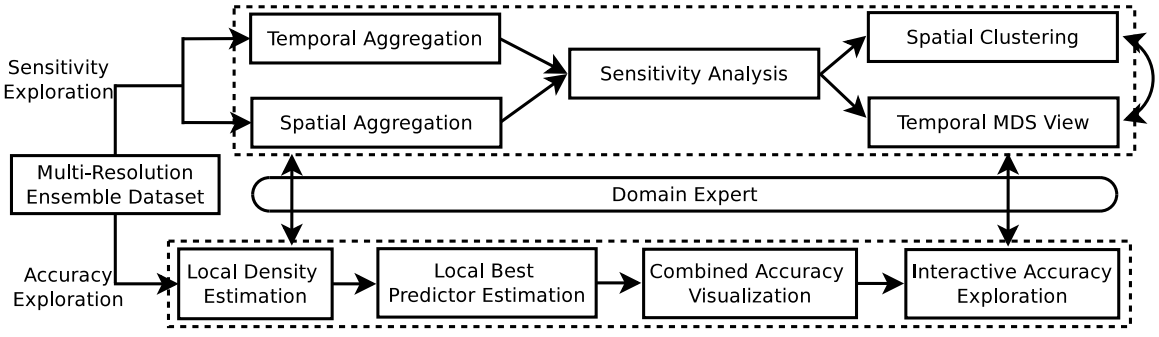


Figure 4.2: A schematic overview of our algorithm.

Our approach is focused on visualization-based exploration of multi-resolution temporal ensemble datasets. Our system design addresses the needs of the domain experts, and the visualization components used in this system are targeted towards answering the previously mentioned queries of the domain experts. As shown in Figure 4.2, our system has two modes of visual exploration: sensitivity exploration mode and accuracy analysis mode. In the sensitivity exploration mode, sensitivity visualization can be performed across resolutions by temporal and/or spatial aggregation of the input dataset. In spatial aggregation, users intend to understand the time-varying sensitivities of the input parameters and to explore the similarity of these sensitivity values across resolutions. After performing sensitivity analysis using the δ method [15] for the five input parameters, users are presented with a multi-dimensional scaling (MDS) projection view along with a connected line chart view. This view helps users understand the relationship of the different resolutions and facilitates further user interaction for more detailed exploration. From the temporal view, users can select different days and explore the similarity in sensitivities of the input parameters as a function of spatial locations. Users can spatially explore the sensitivity information by spatial clustering of the sensitivities. For understanding the multi-resolution sensitivity, the clustering agreement is provided to the user using a custom colormap. Using this connected temporal and spatial view, users can effectively explore the sensitivity relationships by interacting with the system components.

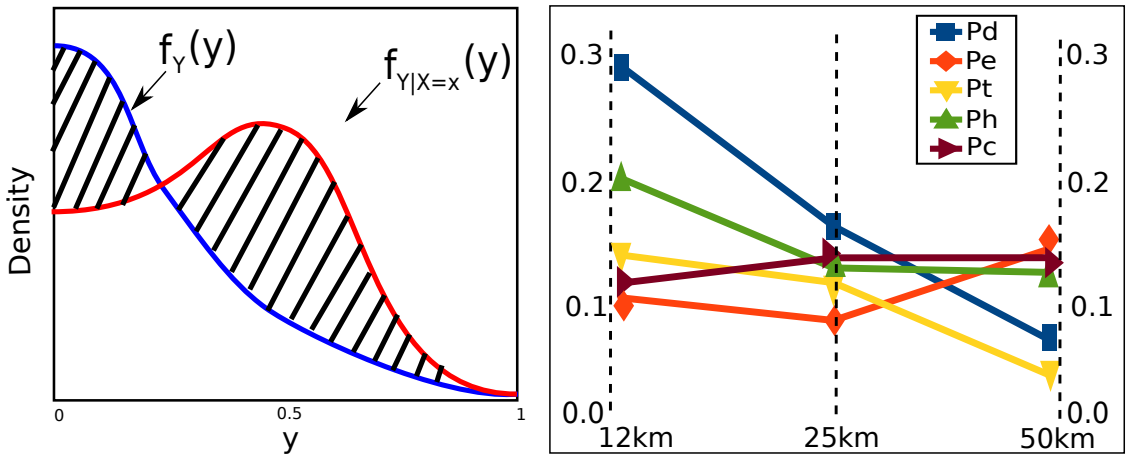
For accuracy exploration, the observed precipitation information is utilized to provide a comparative view of the ensemble outputs at different resolutions. Since every spatial location has a set of ensemble precipitation output for every resolution, initially kernel density estimation is applied to approximate the density functions. Since different resolutions predict the observed precipitation differently, their accuracy is estimated from their

corresponding kernel density estimates. A Bayes rule based method is utilized for this likelihood computation and one of the resolutions is identified as a local best predictor at a spatial location. Since the local predictors can be different over the time domain for a given location, this temporal variation is aggregated to generate an image for combined accuracy visualization. Users can interactively brush and select regions from this image to get information regarding the temporal error on a connected window showing the time-varying error trends. To further explore accuracy on user selected days, spatial error distribution is presented for all the resolutions to facilitate accuracy comparison. An error histogram based selection tool is linked to this multi-resolution view to quickly locate regions of high error. This interactive mode of exploration provides a detailed analysis of accuracy trade-offs across the resolutions.

4.2 Sensitivity Analysis of Input Parameters

4.2.1 Global Sensitivity Measure

Sensitivity analysis is an important topic in operations research (OR) where the analysts strive to quantify the relationship between the input and output variables of their models while taking decisions. As discussed in Chapter 2 (in Section 2.5), the sensitivity analysis measures can be broadly classified into two classes: local and global. In general, a good sensitivity analysis measure is expected to be "global, quantitative, and model free" [113]. Here, "model free" refers to the fact that the measure should not depend on any assumptions regarding the relationships between the inputs and outputs of the model. For the WRF model and its input parameters, a local sensitivity analysis was previously conducted [148] assuming the input parameters were uncorrelated. In this chapter, we use a sensitivity measure named δ that is both "global" and "model free". The δ method was proposed



(a) Sensitivity analysis measure δ computes the shift in the distribution of output variable Y when one parameter is fixed (image courtesy [15]).

(b) Sensitivity analysis measure δ computed at three resolutions of the spatio-temporal averaged data for the five input parameters.

Figure 4.3: Schematic explanation of the δ method and its application on the aggregated data.

by Borgonovo [15]. Let us assume that the model under consideration is $g(\vec{X}) = Y$, where $\vec{X} = \{X_1, X_2, ..., X_n\} \in R^n$, the n uncertain input parameters, and Y is the model output. If $f(\cdot)$ represents the density function, then $f_Y(y)$ is the output density function when all the input parameters are allowed to vary unconditionally. Now, if the input parameter $X_i = x_i$, then the shift in the output distribution is given by the total area calculated as:

$$s(X_i, Y) = \int |f_Y(y) - f_{Y|X_i=x_i}(y)| dy. \tag{4.1}$$

For the ith input parameter, δ_i is then defined as :

$$\begin{aligned} \delta_i(Y) &= \frac{1}{2} E_{X_i}[s(X_i, Y)] \\ &= \frac{1}{2} \int f_{X_i}(x_i) \int |f_Y(y) - f_{Y|X_i=x_i}(y)| dy dx_i. \end{aligned} \tag{4.2}$$

Thus a higher value of δ_i denotes that the expected shift in the density function of the output is higher due to the input X_i, establishing the output to be more sensitive to X_i.

Conversely, if the input parameter X_i has no influence on the output Y, there is no change in the density functions $f_Y(y)$ and $f_{Y|X_i=x_i}(y), \forall x_i \in X_i$. This results in $\delta = 0$ as $s(X_i) = 0$, since $f_Y(y) - f_{Y|X_i=x_i}(y) = 0, \forall x_i \in X_i$. The concept of δ measure is schematically presented in the Figure 4.3(a). This global importance indicator is moment independent since it captures the change in output distribution rather than the change in the variance. As explained by Borgonovo [15], this measure has the property: $0 \leq \delta_i \leq 1$. We use this sensitivity measure for exploration of the weather ensembles as described in the following sections.

4.2.2 Exploration of Parameter Sensitivity Across Resolutions

Given the WRF model generated precipitation dataset, the domain experts want to investigate the sensitivity of the input parameters and how the sensitivity values change when the model runs at different spatial locations and at different days across the three resolutions. The domain experts can choose to explore the sensitivities at different levels of granularity: the finest level being observing the sensitivity of the input parameters at every grid location for every day, the coarsest level is to observe how the input parameters influence the monthly aggregated precipitation over the whole spatial domain. Initially, to provide a high-level view of the sensitivity of the parameters, we first aggregate the dataset to find the most influential parameters and present that to the domain experts. We compute the spatiotemporal average of the precipitation values for each resolution. Next, the δ method is applied and the results are shown in Figure 4.3(b). This shows that Pd is the most important at 12km and 25km but Pe becomes the most important at 50km resolution. Starting from this information, to further understand how the parameter sensitivity changes spatiotemporally, we design two ways for the exploration of the sensitivity values: a) spatial exploration and b) temporal exploration. In the following sections, the time-varying

ensemble precipitation output is denoted as $Y(t,\vec{v},r)$ where t represents days and $\vec{v} \in \{x,y\}$ represents spatial locations, and $r \in \{h,m,l\}$ denotes the different resolutions where h,m,l are abbreviations for *high*, *mid* and *low* respectively.

Spatial Exploration of Parameter Sensitivity

Understanding sensitivity as a function of the spatial locations provides an important insight into the selection of parameters based on locations. Moreover, when simulation models are used to generate data at varying resolutions, the sensitivities of parameters can change across resolutions. Given a period of time, understanding how location-based sensitivity values are comparable across resolutions is of great interest to the domain experts. In addition, weather analysts and domain experts need to apply averaging and smoothing at various levels temporally and spatially. Adhering to this need, after understanding the high level importance of the input parameters as presented in Figure 4.3(b), we investigate the parameters into a finer detailed level by temporally aggregating the ensemble dataset over a user specified θ days. The ensemble output $Y(t,\vec{v},r)$ then reduces to $Y_\theta(\vec{v},r)$ as:

$$Y_\theta(\vec{v},r) = \frac{\sum_{t=1}^{\theta} w_t * Y(t,\vec{v},r)}{\sum_{t=1}^{\theta} w_t}. \tag{4.3}$$

Here t represents days and $\vec{v} \in \{x,y\}$ represents spatial locations, and $r \in \{h,m,l\}$ denotes the different resolutions. The aggregation weights w_t's are domain specific and dependent on the domain expert's choice. The equal weighted w_t's can be used for initial data exploration purposes. Next, for a given resolution r, the importance measure δ is computed at every spatial location $\vec{v}$ for all the input parameters that produces sensitivity value tuples $\vec{\delta}$ at every spatial location, i.e., at every location $\vec{v}$ we have:

$$\vec{\delta}(Y_\theta(\vec{v},r)) \text{ with } \vec{\delta} = \{\delta_i\},\ i = \{1,...,5\}, \tag{4.4}$$

where i represents the different input parameters. After this stage, every spatial location now contains a five dimensional vector describing the sensitivity information of the input five parameters. To understand how these spatial locations are similar to each other according to their sensitivity vectors in a local region, a spatial clustering is applied. To achieve this spatial clustering, we use the k-means algorithm which is popular for its runtime and quality of clustering trade-off. Using this method, the feature vector $\vec{fv}$ consists of the sensitivity values of each location and the location information, i.e., for a given location $\vec{v} = (x, y)$ if the sensitivity tuple $\vec{\delta}$ is computed by following Equation 4.4, then we get $\vec{fv}$ as :

$$\vec{fv} = \{\vec{\delta}, w_{sp} * \vec{v}\}, \text{ with } 0 < w_{sp} < 1. \tag{4.5}$$

The spatial weight w_{sp} is a trade-off between how much the clustering is influenced by the spatial locations compared to the sensitivity values. If $w_{sp} = 0$, then this produces clustering of sensitivity values that are not influenced by the spatial locations.

Since our goal in this chapter is to compare the clustering results across resolutions, instead of computing clustering results for every resolution separately, we incorporate the information from all the resolutions together and then generate the clustering results. We concatenate the N data points from each of the three resolutions to form an input matrix M of dimension $(3N \times \left|\vec{fv}\right|)$. Since in our case $\left|\vec{fv}\right| = 7$, dimension of M is $(3N \times 7)$. Given k clusters as the input parameter to the k-means algorithm, now the individual spatial locations at each resolution can be assigned a color based on the output cluster id. This yields three images for the three resolutions that can be directly compared visually to understand how the sensitivity values agree both spatially and across resolutions.

The clustering results of 12km, 25km and 50km resolutions are presented in Figure 4.4a-c where number of clusters $k = 4$ with $w_j = 0.009$ as the spatial weight factor and

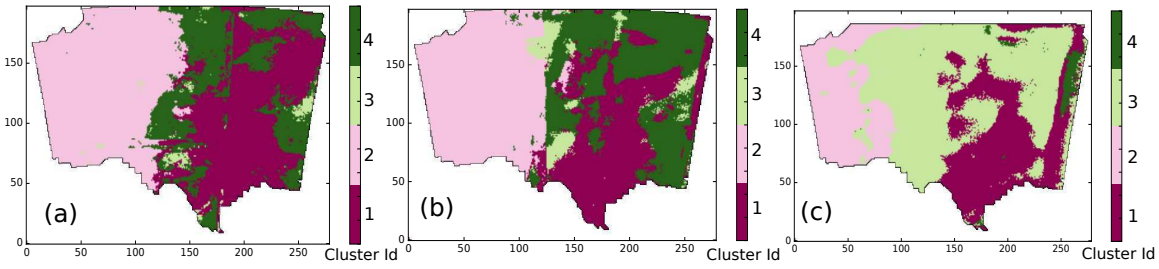


Figure 4.4: Sensitivity-based clustering for three resolutions (a) 12km, (b) 25km, and (c) 50km.

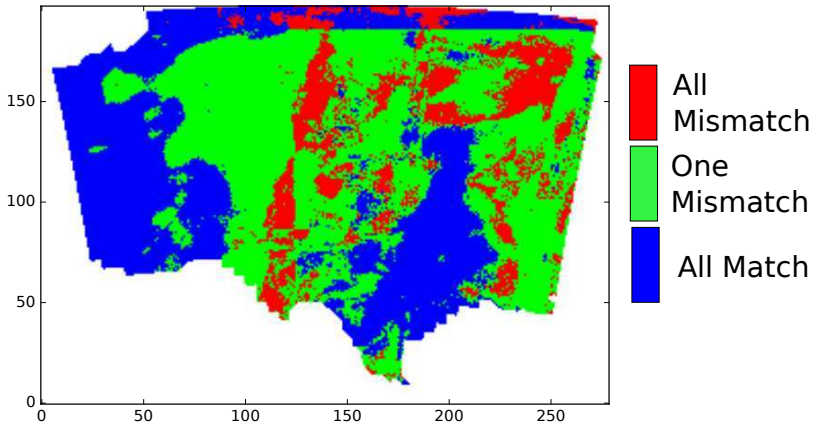


Figure 4.5: The agreement among the three resolutions for the clustering presented in Figure 4.4.

$\theta = 30$ representing a monthly average. It is visually apparent that the simulations at 12km and 25km resolutions show quite similar clustering of the input parameter sensitivities. In comparison, the simulation run at the lowest resolution 50km is much different from the other two. To convey more information through the visualization, the users can select a cluster and analyze the sensitivity of those regions. This helps the domain experts to observe how parameter sensitivities change across regions and resolutions to understand their simulations further.

Finally we provide a summarized clustering information to depict the agreement among the three resolutions. Each location $\vec{v}$ now contains three cluster ids $\{id_h, id_m, id_l\}$ originating from three resolutions. We present a combined visualization of the three clustering images from three resolutions to depict the agreement of these resolutions in an integrated view. Given $\vec{id} = \{id_h, id_m, id_l\}$ at every spatial location (x, y), a color conversion function $Cmap(\vec{id})$ is defined as:

$$Cmap(\vec{id}) = \begin{cases} blue, & \text{if } id_h = id_m = id_l \\ red, & \text{if } id_h \neq id_m \neq id_l \\ green, & \text{otherwise} \end{cases}$$

The agreement of the clustering information provided in Figure 4.4a-c is shown in Figure 4.5. The blue color shows the region where all the three resolutions show similar clustering, red color shows the regions where all the resolutions mismatch and green shows where exactly two of the three resolutions match but the other one does not. This figure shows that in the regions where all the three resolutions predict high rain or very low rain, the sensitivities match. But for the regions where there are some variations in the predicted output precipitation, the sensitivity values also differ. This exploration motivates the users to further investigate their simulation parameters.

Temporal Exploration of Parameter Sensitivity

After exploring the sensitivity of input parameters spatially, our next goal is to understand the parameter importance in time-domain for all the three resolutions. The domain experts need to understand the sensitivity of the parameters on a daily basis or over a specific period of time. While making predictions regarding the precipitation of a given period, understanding which input parameters have the most influence is of prime importance. Understanding the progression of sensitivity values over time leads to a better optimization of parameter space sampling and faster convergence with better predictions. To address

this domain need, in our design we assume an aggregation period of α days and we get the time-varying data $Y_\alpha(t,r)$ by computing the spatiotemporal average behavior of the ensemble precipitation dataset as:

$$Y_\alpha(t,r) = \frac{\sum_{k=t}^{t+\alpha} w_k * \sum_{\vec{v}} Y(k,\vec{v},r)}{\sum_{k=t}^{t+\alpha} w_i},\ t = 1,...,D-\alpha+1. \tag{4.6}$$

Here t represents time, r represents a given resolution and D is a constant representing the total number of days for which the simulation is run. The aggregation weights w_k's are again dependent on the simulation model. It can be readily observed that the Equation 4.6 will reduce to daily temporal analysis when $\alpha = 0$. When $\alpha = D$, it will reduce to the monthly spatiotemporal aggregation. For any other choices of α where $0 < \alpha < D$, $Y_\alpha(t,r)$ is the moving average of $\sum_{\vec{v}} Y(t,\vec{v},r)$ in temporal dimension. Now the importance measure δ is computed at $t, \forall t = 1,...,D-\alpha+1$ that produces $\vec{\delta}$ as:

$$\vec{\delta}(Y_\alpha(t,r)) \text{ where } \vec{\delta} = \{\delta_i\},\ i = \{1,...,5\}. \tag{4.7}$$

For a systematic exploration of temporal sensitivity, we design a visualization component that depicts the information contained in $\delta(Y_\alpha(t,r))$. Since each day is now represented by a five-dimensional feature vector $\{\delta_i\}$, and there are three different resolutions, a data transformation is required when the users want to explore the similarities of different resolutions over $D-\alpha+1$ days. To meet this need, we first apply dimensionality reduction technique to project the five-dimensional data points to two-dimensions by preserving the higher dimensional distances among the points as much as possible. Since each day is represented by a five-dimensional feature vector for each of the three resolutions, there are total $3 \times (D-\alpha+1)$ such high-dimensional points. We employ the popular multi-dimensional scaling (MDS) technique [30] to reduce the dimensionality of our input set of data points from five dimensions to two dimensions. MDS is popular since it is a non-linear

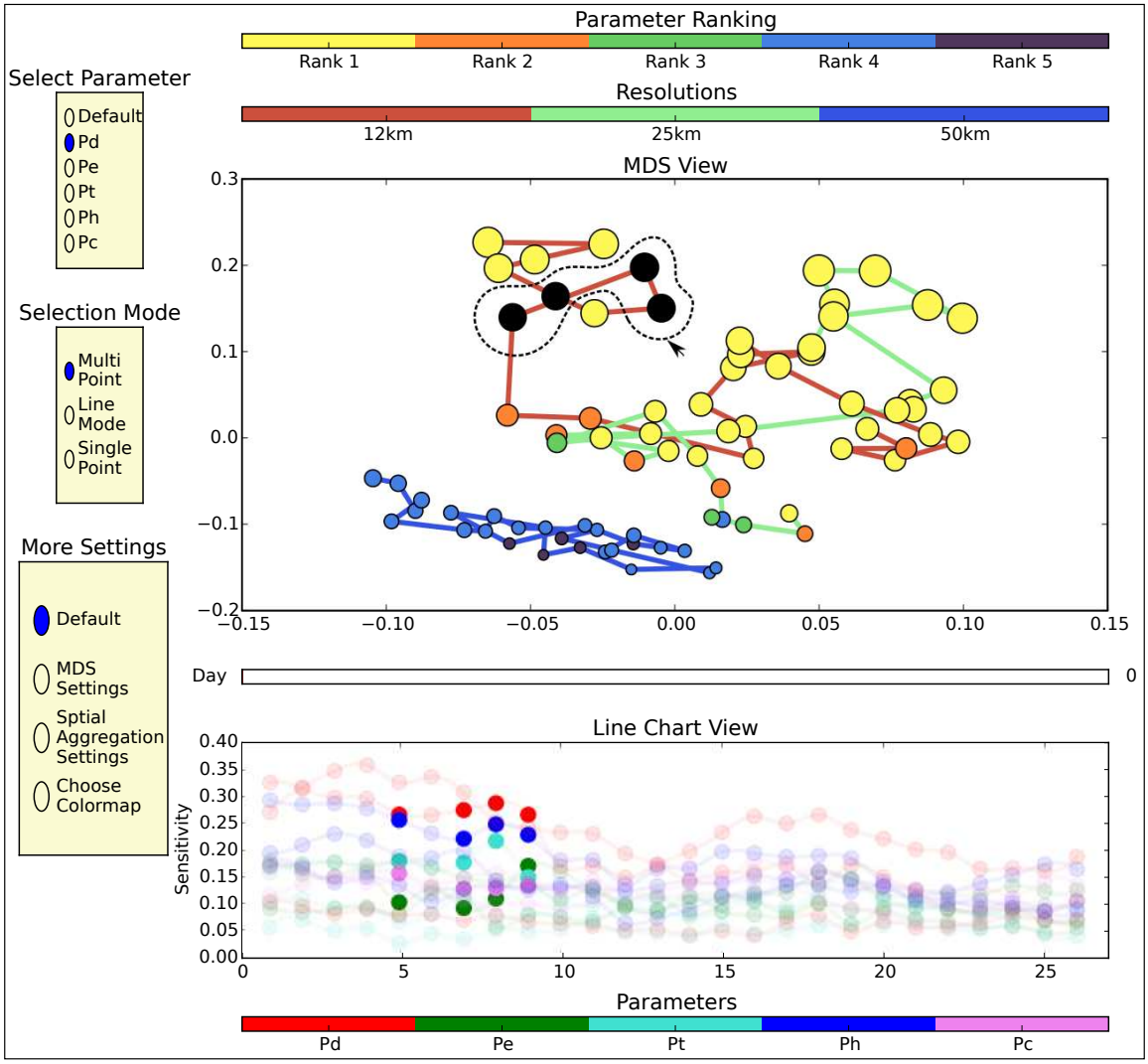


Figure 4.6: Time-varying sensitivity visualization using our proposed technique. Top: the MDS plot shows the similarity among the different days across three resolutions where each point is a five dimensional vector of sensitivity values of the five parameters. Bottom: Line chart shows the detailed sensitivity values for the user selected days.

dimensionality reduction technique and also it preserves the high dimensional distances in the projected lower dimension.

Application of MDS generates a scatter plot in two-dimensional space for the different days of the three resolutions as shown in Figure 4.6. In this figure, we have used the time

aggregation period $\alpha = 5$. The points from the same resolution are connected by straight line segments where the color of the line segments designate the particular resolution (red for 12km, green for 25km, blue for 50km). From this MDS plot, it can be readily inferred that the lowest resolution behaves very differently compared to the other two resolutions when all the parameter sensitivity values are considered together. To show the progression of time, a "Day" slider tool-bar is used that can be tuned by the users to highlight the corresponding day in the MDS plot from all the three resolutions. From this MDS view, users can also compare the sensitivity values of the different days within and across resolutions. Since domain experts are interested in observing which parameter is the most important and how much important, we use the size and color of the MDS plots to convey these two pieces of information. For a selected parameter for each day and resolution, its rank is computed in comparison to the other parameters and it is color-mapped in the scatter plot according to a predefined discrete color-map. The absolute sensitivity score of the selected parameter is used to modulate the size of the scatter plot points. Using this design, more important parameters will be highlighted in larger size and brighter colors for the different days and resolutions. In Figure 4.6, parameter Pd is selected to demonstrate this design. As can be seen, Pd is very important in the 12km and 25km in deciding the precipitation, but at the lowest resolution 50km, Pd becomes less important.

After observing the high-level view of the temporal trend, users can opt for detailed analysis of the individual days. Using a lasso-based selection tool, users can select days from different resolutions and simultaneously view them in a connected line chart. This line chart contains the time-curve of each parameter from all the three resolutions plotted together. Based on user's selection, all the parameters from the selected days and resolutions are highlighted in the line chart. The MDS view and the line chart view are also connected

to the spatial views (discussed in Section 4.2.2) for detailed spatiotemporal analysis of the sensitivity values.

4.3 Accuracy Analysis Across Resolutions

After analyzing the sensitivity of the input parameters of the simulation model, another important part is to understand the accuracy of the simulated outputs. Since the ensemble datasets are generated at different resolutions, understanding the effect of the resolution is of great interest to the domain experts. Although it is generally expected that the higher resolution simulation output will be able to predict the observed phenomenon much better than the lowest resolution, it is not necessarily true in the real ensemble simulations. With the increase in resolution, the model uncertainty can also increase for a simulation like the precipitation of WRF model. Domain experts need to understand these variations and intend to get information for future adaptive grid implementations. In this section, we discuss our proposed solution to handle this scenario for helping the domain expert with an effective exploration tool.

4.3.1 Bayesian Model for Accuracy Comparison

Given an ensemble dataset and its observed ground truth, Bayesian Model Averaging (BMA) [47] was previously employed to understand the accuracy of the multiple ensemble runs. Since the dataset we are dealing with is multi-resolution, we formulate a different strategy for accuracy analysis. We use a Bayesian approach across the three resolutions to find the estimated best predictor. Given a time t and location $\vec{v}$, our simulation model generates a collection of precipitation estimates for every resolution. Let d_i represent the distribution of the rainfall values generated by the ith resolution at a given location at a

given time and R is the random variable for the estimated rainfall values. Let D be the random variable that represents the user's choice of which distribution d_i is selected from the k distributions originating from the k resolutions. Thus, $P(D = d_i)$ represents the probability of choosing the ith distribution from the k distributions. Now, at a given location with the observed rainfall amount r, we want to identify the best local predictor as the ith resolution when the conditional probability $P(D = d_i|R = r)$ is maximized:

$$\underset{i \in k}{\operatorname{argmax}} \; P(D = d_i|R = r). \tag{4.8}$$

From the Bayesian formulation, it can be readily observed that:

$$P(D = d_i|R = r) \propto P(R = r|D = d_i)P(D = d_i). \tag{4.9}$$

In this equation, $P(D = d_i)$ is the prior unconditional probability of choosing ith resolution as the best predictor. In the absence of the prior information, $P(D = d_i)$ can be set to equally probable for all the resolutions, i.e., $P(D = d_i) = \frac{1}{k}$. Depending on domain expert's choice, this probability can be set to be inversely proportional to the computation and storage cost of the ith resolution dataset, i.e., $P(D = d_i) \propto Cost(i) \Rightarrow P(D = d_i) = \varepsilon * Cost(i)$. Domain experts can tune this prior probability for understanding the cost vs accuracy benefits of multiple resolutions.

4.3.2 Kernel Density Estimation for Ensembles

The conditional probability $P(R = r|D = d_i)$ of Equation 4.9 intuitively captures the likelihood of distribution d_i generating the sample r. Although Gaussian assumption is popular among the weather analysts, in our method we take up a non-parametric approach and use kernel density estimation (KDE) to approximate the distribution of the ensemble outputs at every spatial location. Although slower, this KDE approach is beneficial over

the Gaussian approach since 1) it is parameter-free unlike Gaussian estimation, 2) Gaussian distributions are known to produce non-negligible likelihood of getting negative values whereas our precipitation data is strictly positive. For KDE, the popular choices of kernel types include Gaussian, triangular, rectangular, and the Epanechnikov kernel. Different choices of the kernel types produce different estimated density functions although the variation due to kernel types is considered less significant compared to the choice of kernel bandwidth [106, 125]. When the Gaussian kernel is used, Silverman's rule of thumb for bandwidth selection produces a good quality density estimation. Following this method, bandwidth h is given as $h = (\frac{4}{d+2})^{1/(d+4)}\sigma n^{-1/(d+4)} = 1.05\sigma n^{-1/5}$ where σ is the standard deviation of the ensembles at a given location, n is the number of ensemble runs and d is the data dimensionality ($d = 1$ in our case). The kernel was chosen to be the Gaussian kernel with this method of bandwidth selection.

Using the KDE approach at every spatial location, three distributions can be estimated from the ensembles coming from the three resolutions. From these density functions, the local best predictor is identified by approximating the conditional probability $P(R = r|D = d_i)$ as the likelihood of r from the corresponding density functions. Thus, $P(D = d_i|R = r)$ can be approximated as Equation 4.9 and comparing the values from three resolutions, the best local predictor can be identified. Since the densities estimated from KDE are used for comparison, the same bandwidth needs to be used for a spatial location across all the three resolutions while computing the d_i's. After identifying the local best predictors for each time step, our goal is to explore the prediction performance of the three resolutions across time domain for different spatial locations. To achieve this, we find the three tuple $\{e_h, e_m, e_l\}$ that represents the temporal prediction performance of the three resolutions at a given location. Here e_h denotes the count of how many times the highest resolution

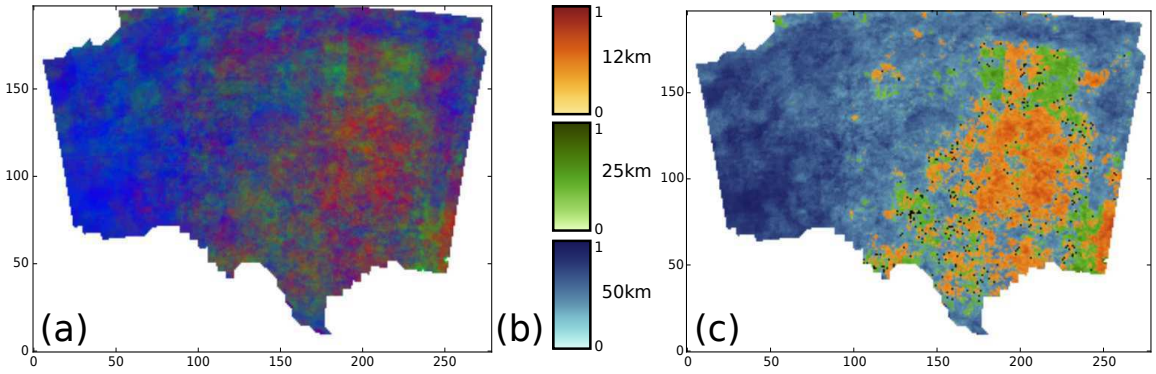


Figure 4.7: Combined resolution map for accuracy exploration. (a) Simple rgb color-mapped visualization is ineffective for understanding the dataset. (b) Our proposed method of first classifying and then mapping to color is more useful for exploration of the dataset.

was chosen as the best predictor at the given location over all the days; similarly, e_m and e_l denote the counts for middle and lowest resolutions respectively. Finally, we turn the counts into corresponding probability values $\{p_h, p_m, p_l\}$ by dividing the counts by the total number of days. Thus, p_h now represents the probability of selecting the highest resolution as the best predictor over all the days at the given location. After this conversion from the ensemble data to probability values, the resulting dataset can be used for visualization.

4.3.3 Visualization of Multi-resolution Efficiency

To use the previously generated probability data, a visualization is needed that easily and effectively transfers the information to the domain expert. Since there is a tuple $\vec{p} = \{p_h, p_m, p_l\}$ at every location of a two-dimensional field, one option is to directly map the probability values to r, g, b values for visualization. But this produces an ineffective visualization as shown Figure 4.7a. Instead, we first assign three disjoint colormaps to represent the three resolutions and depending on which resolution has the highest probability at a given location, the final color is selected from the associated colormap. In essence, we

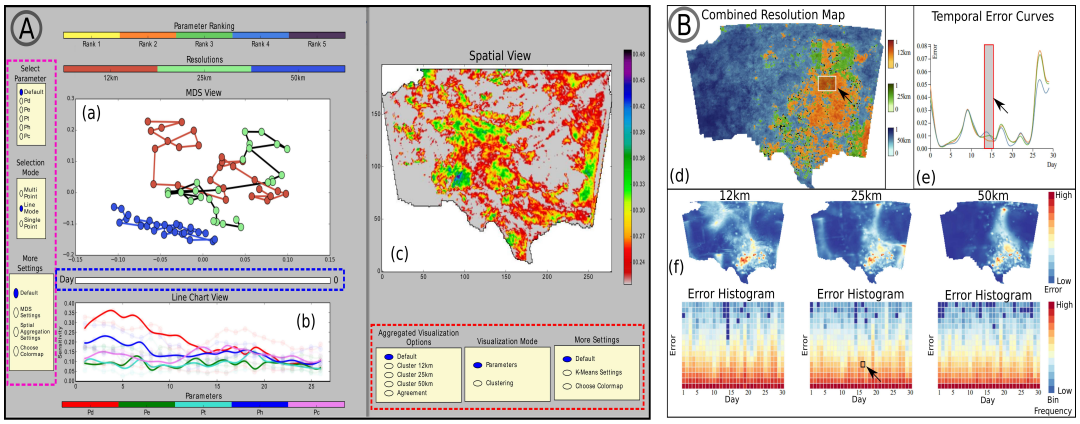


Figure 4.8: Visualization of our system: (A) sensitivity exploration mode, (B) accuracy analysis mode.

first classify each location based on the probability values and then use the largest probability value to modulate the color. Since different colors have different brightness to the human eye and their contrasts are perceived differently, special care is needed in choosing the colormaps. In this work, we collaborated with an artist who has more than twenty years of experience in the art and visualization domain to select the three colormaps for the three resolutions. These three colormaps are presented in Figure 4.7b and using this custom color coding, final output is generated as shown in Figure 4.7c. From this aggregated image, domain experts get an idea of where the different resolutions are performing better than the others. Comparing this image with the mean observed data as shown in Figure 4.1(c), it is readily observed that the highest resolution ensemble performs quite well in the regions of higher rainfall. Similarly, the lowest resolution predicts the dry regions better than the other two resolutions.

4.4 System Integration and User Interaction

In this section, we integrate all the system components together to help the users explore the time-varying multi-resolution ensemble datasets. Here we discuss some key concepts regarding our system. A snapshot of the system for sensitivity analysis is shown in Figure 4.8A where we have three connected visualization windows, Window (a),(b),(c). In Window (a), the MDS plot of the spatially aggregated data points (representing different days) are displayed as discussed in Section 4.2.2. In the "Default" view of the "Select-Parameter" option of the selection panel (marked by violet outline), the MDS points are color-mapped to represent the different resolutions to provide a high-level view of how the three resolutions are related over the temporal domain. From the selection panel of "Select-Parameter", users can select an input parameter to explore the rank and sensitivity of it across all the three resolutions similar to Figure 4.6. In this mode, the point sizes are scaled according to the sensitivity values of the selected parameter and the color of points reflect the rank (brighter colors represent higher ranks according to a custom colormap shown at the top of the window (A)). Users are provided with a "Day" slider (marked by the blue outline) to interactively highlight different points of the MDS plot across all the resolutions for sensitivity-based similarity comparison. From the left selection panel, users can operate in "Multi-Point" mode to select any number of days by drawing a free-hand curve around the points. The line chart view of Window (b) will then be updated to show the sensitivity values of all the five parameters for the user-selected days. When in "Line-Mode", users can select one of the three resolutions from the MDS view to observe the temporal sensitivity trends of the five parameters as time-curves in the line chart window. In Figure 4.8, 25km resolution is selected and the sensitivity trends are shown in Window (b). From the "Single-Point" mode, users can select a day from Window (a) for spatial exploration

of the selected parameter in Window (c). For spatial view, Window (c) has two modes of visualization: 1) parameter sensitivity distribution across spatial locations, 2) clustering information for different spatial locations. When the "Parameters" view is selected, it shows the spatial distribution of δ values of the selected parameter for the user selected day. In this mode, the users can also interactively set the colormap of the visualization. When the "Clustering" view is selected, it shows how the clustering based on the input parameter sensitivities (as discussed in Section 4.2.2) vary across spatial regions for the user selected day. To provide a time-aggregated view of the clustering, the user can also select to view the clustering information of the temporally averaged data for the three resolutions similar to Figure 4.4. In this interface, users can alter the system settings (shown in red and violet outlines) and continue to explore in detail how the sensitivity values of the input parameters change across spatial locations, temporal domain, and three resolutions.

In the accuracy exploration mode, as shown in Figure 4.8B, we first present the combined resolution view of the dataset in Window (d) using our custom colormap (method discussed in Section 4.3). Users can interactively use brushing-and-linking to explore the regions of Window (d) for further analysis. Since our combined map view only presents which resolution is performing better at which spatial locations, we add the error analysis to augment the visualization. In this case, we are using the popular root-mean-square-error (RMSE) measure for each point in the spatiotemporal domain. Further, the RMSE errors are spatially aggregated for each day to generate "temporal error curves" for each of the resolutions in Window (e). These simultaneous curves show a comparative view of how much error each resolution produces in predicting the observed rainfall over the temporal domain. When a rectangular region from Window (d) is selected (marked with a white box), the error curves of Window (e) are updated to show the temporal trend of error for

the user-selected region. Users can also select an interval of days from Window (e) (marked by red box) and further explore how the error is distributed over the spatial domain for the three resolutions for those user-selected days in the connected spatial-error views provided at the top panel of the Window (f). For the users to quickly identify the regions of high error from the three resolutions, we provide an error histogram panel below the spatial-error view of the individual resolutions of Window (f). In these error histograms, the RMSE errors of each day are separated into user-defined k-bins. The x-axis represents the different days and the y-axis shows the increasing error values. The height of each bin is mapped to color according to a user-defined color-map. This setting helps the users identify the high frequency and/or high error bins easily. Users can select an error bin (marked with the black box) from one resolution, and the spatial locations contributing to that bin are highlighted in the connected spatial-error map view. The error generated in the other two resolutions for these spatial locations are also highlighted in the corresponding spatial views. Using these interactions and methodology, our designed system can provide the users a comprehensive view of the accuracy and error in the simulation generated output across multiple resolutions.

4.5 Domain Expert Feedback

We collaborated with an expert of the field with multiple years of experience in this domain. Our proposed system was developed after a series of thorough and detailed discussions with the expert. The domain specific requirements were communicated in multiple phases, thereby refining the tools and methods that are to be used in the final software. The domain expert was involved at every stage of the pipeline to ensure that the goals of the experiments were achieved. Expert feedback was obtained during the several meetings that

we organized with the domain expert throughout the development phase and after the tool was completed. The final comments of the expert are listed here that point to the benefits of our system and future improvements.

4.5.1 Feedback on the System and Methodology

The feedback we received from the expert was very encouraging. The expert commented that our designed tool enabled him to explore the sensitivity information at different levels of data aggregation. This is useful to him especially because a lot of averaging is done by the experts on the weather simulation output to predict and understand the climate situations over different time and spatial domains. Our proposed tool provided the methodology to walk through the different data aggregations with simple user inputs and interactions. The expert did not have an existing visualization tool that explores these aspects of the input parameters. He also mentioned that the accuracy analysis tool effectively depicts the performance of the different resolutions with simple user interactions. Our system generated images clearly establish that the highest resolution simulation does not always perform better than the lower resolutions, which is counter-intuitive but important knowledge for the domain expert. Since our tool effectively highlights the spatial and temporal behavioral patterns of different regions, the expert was able to quickly locate the days and regions of interest from the tool. Since this software was cross-domain, we kept the interactions easy and intuitive with intuitive color-maps. The ease of use was another aspect that the domain expert agreed on. Overall, the expert expressed that the system met the goals for the project and all the components used in the system were needed for the system to perform well.

4.5.2 Domain Specific Discoveries

Our system helped the expert in making several domain specific discoveries. The MDS view revealed that the lowest resolution has very different sensitivity values on the different days compared to the other two higher resolutions. The MDS view connected with the line charts also reveals that the input parameter *Pd* which is the coefficient related to downdraft mass flux rate is the most important input parameter in determining the output precipitation for most of the days of high and middle resolutions. For the lowest resolution, all the input parameters are very similar with low sensitivity values. This was a new discovery for the dataset which was revealed due to the detailed sensitivity analysis. The clustering based spatial view revealed that for the overall days, the different resolutions agree with the sensitivity of the output precipitation to the input parameters for the regions that are either very wet or very dry. Also, from individual clustering visualizations, it was revealed that the lowest resolution was again very different from the two higher resolutions in the spatial clustering information as well. Similarly, in the accuracy exploration, the fact that different regions are well predicted by different resolutions was very crucial information for the domain expert. Although it was expected that the higher resolution may not always provide better prediction compared to the lower resolutions, the performance of the lowest resolution in the less wet and dry regions compared to the other two resolutions, was surprising to the expert. The domain expert later concluded that the reason behind this observation is the model uncertainty contained in the WRF climate model. Due to this, when increasing the model resolution, the model may become more unstable and cause larger model uncertainty. This may result in higher inaccuracy for the higher resolution in certain areas compared to the lower one. Also, our expert expressed that the knowledge of high

resolution as a better predictor for the more wet regions will guide the expert towards a better adaptive grid refinement in the future.

4.5.3 Future Improvements and Updates

The domain expert also provided us with suggestions for improving our system. Instead of incorporating only the δ measure for sensitivity analysis, the expert wanted us to include more sensitivity measures to understand how they rank the different parameters. Understanding the consensus among the parameters can help us predict the parameter sensitivity more robustly. In the accuracy analysis, the domain expert wanted us to develop an AMR grid based interactive cost optimization strategy that can be user controlled. Here the cost refers to the computational cost of the model. The domain expert also wanted us to extend our visual exploration system to *in situ* workflow and also to handle three-dimensional multi-resolution temporal ensembles. Finally, we plan to involve multiple domain experts for further improvements and feedback on our system. We take these suggestions from the domain expert as our future work.

4.6 Conclusions

In this chapter, we presented a complete system to explore multi-resolution temporal weather ensembles. Our system is intended for exploring the sensitivity of the input parameters and accuracy of the output precipitation across resolutions. In our proposed method, we used a global sensitivity measure named δ to compute the sensitivity of the different input parameters over spatial regions and over temporal domain for the three given resolutions. To explore how the sensitivity of the parameters change over spatial regions, a location-based clustering is applied for different days and the clustering-agreement among the three resolutions is computed. For analyzing temporal patterns, spatially aggregated

dataset is used for sensitivity computation given all the parameters and an MDS plot linked with line charts is presented for user interaction. For accuracy exploration, a Bayes rule based likelihood method is used to compute the local best predictor among the three resolutions for every spatial location from the ensembles. To create local probability density functions for this likelihood comparison, KDE approach with Gaussian kernel is used. The local best predictors are then aggregated in the temporal domain to find which location is better predicted by which resolution over the total number of days and it is color-mapped for further user interaction. Users can now select regions from this image and analyze the temporal and spatial error pattern in a linked window. We carried out the experiments in close collaboration with an expert whose feedback establishes the efficacy of our proposed system.

Chapter 5: Uncertainty-Driven Vortex Detection in Flow Fields

Compared to the relationship analysis discussed in the previous two chapters, in this chapter we discuss the detection and analysis of uncertain features; more specifically vortices. Vortices are one of the most important features in the flow fields and robust detection of vortices is a very popular research topic for the scientists. For vortex detection, uncertainty analysis was previously conducted using probabilistic methods assuming the data contained uncertainty [96]. In this chapter, we propose a method to analyze the uncertainty that arises from the differences in the performance of existing vortex detection schemes, which act as inconsistent oracles.

A number of the existing detectors are point-based and are called local detectors. These detectors classify points in the field by assigning membership to either the class vortex or the class non-vortex based on a local point-wise criterion. Typically, they use a hard-threshold for the classification purposes and the determination of an optimal threshold is necessary for robust detection [25, 38, 153]. On the other hand, non-local vortex detection schemes analyze a region and classify it based on the absence or presence of a vortex. The use of streamlines is very prevalent in this class of detection algorithms. Generally, these algorithms require significant user intervention to perform well. Despite these efforts [74], there does not exist a robust and reliable method that detects vortex structures with full certainty. Thus, there exists a mismatch among the regions identified as vortices by

the different detectors. Moreover, the threshold values used by these different detectors provide a reference related to the degree of confidence of the prediction. If the output from a detector is very close to its theoretical threshold, then the prediction contains more uncertainty. This observation enables us to model this uncertainty as a fuzzy membership problem for the vortex and non-vortex classes in the flow field. To reduce the uncertainty in the detection process, a second characteristic that we exploit is *locality*. A vortex is defined as a contiguous region within the domain. A study of vortical features reveals that a point has a higher chance of being part of a vortex if it is located near pre-identified regions of a vortex while a point that is not located near a vortex is much less likely to be part of a vortex. The inclusion of a well-formed distance-criterion into these vortex detectors can efficiently reduce their uncertainty of prediction.

In this chapter, we introduce a novel uncertainty-driven vortex analysis workflow that enables the users to analyze and extract vortices with higher accuracy and robustness. We use four existing local vortex detection techniques as the basis of the identification process. For each of the detectors, we apply a logistic function that converts the output of each detector to a fuzzy *possibility value* in the range of 0 to 1. Using the method of entropy maximization, we allow for parameters to be automatically selected for this conversion. The possibility values generated at this stage are used for a voting scheme to identify the regions of a candidate vortex structure with a higher degree of certainty. Utilizing the property of spatial locality, uncertain points are then classified into either vortex and non-vortex classes. In this workflow, a domain expert provides her/his input by marking a small number of sample points, which are used for estimating the parameters of the system based on global information. To the best of our knowledge, our approach is the first attempt to incorporate spatial locality with uncertainty analysis for the detection of vortical features.

Our contributions are threefold in this chapter:

1. We introduce a new approach towards modeling the uncertainty of the existing vortex detectors by using their threshold values to create a fuzzy input-based system.

2. We present a consensus-based voting method to identify vortex regions with a higher degree of confidence and segregate the different certainty levels for different spatial regions.

3. We introduce the use of spatial proximity into the vortex detection framework. Different vortex clusters are identified and a distance-based criterion is used to assign the final vortex points in the system.

5.1 Vortex Detection Methods

5.1.1 Global Streamline Method

Global methods for vortex detection generally attempt to find a coherent region in the field with closed or spiraling streamlines in a reference frame moving with the vortex. An intuitive, streamline-based definition of a vortex was provided by Robinson [107] as: "A vortex exists when instantaneous streamlines mapped onto a plane normal to the vortex core exhibit a roughly circular or spiral pattern when viewed from a reference frame moving with the center of the vortex." This definition encapsulates the inherently global nature of a vortex. Unfortunately, it is self-referential and difficult to describe algorithmically. Determining the appropriate reference frame moving with the vortex is a non-trivial issue [141]. Additionally, seeding the streamlines at the correct location is of utmost importance. Analysis of the geometric properties of a streamline [73, 97] or computation of the winding angle [111] are two examples of streamline-based vortex detection methods.

5.1.2 Local Vortex Detectors

Local vortex detectors evaluate a function at each point of the field and generate a numeric value. This field value is used to decide whether the point is a member of a vortex region by the help of a threshold function. Different local detectors have different threshold functions and the choice of the threshold plays an important role [25, 153, 38] for improving the precision of the vortex detection algorithm. Most of the local vortex detectors are based on the velocity gradient tensor J, where $J = \bigtriangledown v$. For this reason, these detectors are Galilean invariant or invariant under translation. The estimated velocity gradient tensor is used to compute the rate of strain tensor S and the rate of rotation tensor R as follows:

$$S = \frac{1}{2}(J + J^T), R = \frac{1}{2}(J - J^T). \tag{5.1}$$

The four most popular local detectors include the Q-criterion, λ_2, Δ-criterion, and Γ_2. After generating values at each point, a function L can be defined for each of these detectors with $L : \mathbb{R} \longrightarrow \mathbb{B}$, where $\mathbb{B} \in \{0, 1\}$. The function L is used for labeling the flow field points with 1 as vortex label and 0 as non-vortex label. This function depends on the threshold of the detectors and their characteristics.

Q-criterion:

The Q-criterion was proposed by Hunt et al. [62]. It identifies the vortex by finding the regions where the rotation of the field exceeds the strain. In addition, the pressure in this region needs to be lower than the ambient pressure. The Q-criterion is defined as:

$$Q = \frac{1}{2}(\| R \|^2 - \| S \|^2). \tag{5.2}$$

The corresponding labeling function L_Q is defined as:

$$L_Q(x) = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{if } x \leq 0 \end{cases}. \tag{5.3}$$

λ_2 Method:

The λ_2 method was proposed by Jeong and Hussain [72] who postulated that a minimum pressure region is not a sufficient condition for detecting the vortices. They proposed a vortex detection scheme by extracting a connected region wherein the matrix $S^2 + R^2$ has two negative eigenvalues. Assuming that λ_1, λ_2 and λ_3 are the three eigenvalues where $\lambda_1 \geq \lambda_2 \geq \lambda_3$, then if $\lambda_2 < 0$ at a given point in the field, then this point belongs to a vortex core region. This condition corresponds to the existence of a rotation-induced local pressure minimum. Following this method, the labeling function is defined as:

$$L_{\lambda_2}(x) = \begin{cases} 1, & \text{if } x < 0 \\ 0, & \text{if } x \geq 0 \end{cases}. \tag{5.4}$$

Δ-criterion:

Chong et al. [27] defined the Δ-criterion for vortex detection using critical point theory. In this definition, a vortex core is identified as a region where J has complex eigenvalues, which would produce cyclic trajectories in the region near the critical point. The Δ-criterion is defined as:

$$\Delta = \left(\tfrac{1}{2}P\right)^2 + \left(\tfrac{1}{3}Q\right)^3 \tag{5.5}$$

where $P = Det(J)$ and $Q = \frac{S_{ij}S_{ji} + R_{ij}R_{ji}}{2}$. For the vortex labeling, the labeling function for this method is defined as:

$$L_\Delta(x) = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{if } x \leq 0 \end{cases}. \tag{5.6}$$

Γ_2 Method:

Graftieaux et al. [49] proposed the Γ_2 method for detecting the vortex boundary. This method identifies the regions where rotation dominates strain by looking at the maximum eigenvalues of the rotation tensor and strain rate tensor. This method can be implemented by considering the ratio $\| r \| / \| \mu \|$ where $\| r \|$ and $\| \mu \|$ denote the maximum eigenvalues of the rotation tensor matrix R and the strain rate tensor S respectively. The labeling function is given as follows:

$$L_{\Gamma_2}(x) = \begin{cases} 1, & \text{if } x > 1 \\ 0, & \text{if } x \leq 1 \end{cases}. \tag{5.7}$$

5.2 Method

5.2.1 Uncertainty Quantification for Vortex Detectors

In the vortex detection problem, if a pointwise detector's output is close to its theoretical threshold value, then the classification is generally more uncertain compared to the case where the value is further away from the threshold. This is represented schematically in Figure 5.1(a) with th as the threshold value. The hard-thresholding of the pointwise detectors ignores this property and generates binary 0 or 1 as an output where 0 denotes non-vortex regions and 1 denotes the presence of a vortex. To incorporate the uncertainty that is inherent to these vortex detectors, instead of generating hard-thresholded 0 or 1 output from the existing detectors, it is more desired to generate fuzzy outputs in the interval $[0,1]$ such that it represents the *vortexness* of the point given the detector.

Previously Burger et al. [20] chose a linear mapping function to compute the vortexness. In our work, we use the sigmoid function F which is given as:

$$F(x) = \frac{1}{1 + e^{-a(x-th)}} \tag{5.8}$$

where the parameter th denotes the point of maximum uncertainty: when $x = th$, $F(x) = 0.5$. The parameter a represents the fall-off rate or the steepness of the sigmoid curve. We use a sigmoid function for the following reasons: 1) The sigmoid function is more general approach compared to linear mapping. Further, the sigmoid function can be used to imitate the linear mapping through proper choice of the parameters. 2) Classification problems are inherently sigmoidal. To illustrate this point, let us consider two classes C_1 (vortex class) and C_2 (non-vortex class) and an observation x (output of a vortex detector), the posterior probability of classification $p(C_1|x)$ can be written [12] following the Bayesian approach as:

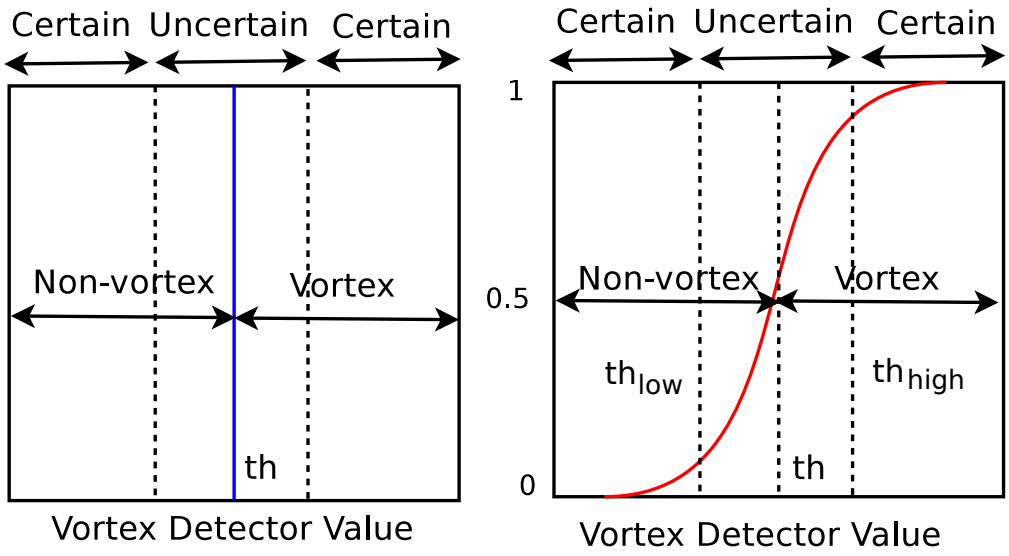
$$p(C_1|x) = \frac{p(x|C_1)p(C_1)}{p(x|C_1)p(C_1) + p(x|C_2)p(C_2)} = \frac{1}{1+e^{-a}} \tag{5.9}$$

where the value of a is defined as $a = \ln \frac{p(x|C_1)p(C_1)}{p(x|C_2)p(C_2)}$. In Equation 5.9, $p(C_1|x)$ follows a sigmoid or S-shaped distribution with a steepness factor a as shown in Figure 5.1(b).

For vortex detection, the sigmoid function F can be applied to the outputs of the individual vortex detectors to convert them to fuzzy *vortexness* values. After the fuzzy conversion, values closer to 1 indicate a higher possibility to be part of a vortex. If the output value is close to 0, then it is more likely to be marked as non-vortex. The values around 0.5 are the uncertain values. Figure 5.1(b) shows a schematic view of this conversion. Given a collection of vortex detectors, we assign one fuzzy conversion function F to each individual detectors by tuning the parameters th and a as discussed in Section 5.2.4.

5.2.2 Uncertainty Reduction Through Aggregation of Multiple Oracles

Since the individual vortex detectors can be inconsistent in their identification of a vortex, they can be treated as the imperfect oracles of our vortex detection system. It has



(a) Certain and Uncertain regions after hard-thresholding around th.

(b) Uncertainty in the output is modeled using a sigmoid curve.

Figure 5.1: Uncertainty in vortex detection and modeling via sigmoid curve.

been shown [128, 132, 153] that the use of multiple detectors instead of a single detector, provides a higher level of reliability for vortex detection and visualization. After converting the outputs of the individual vortex detectors to represent their fuzzy confidence levels, we propose to aggregate their prediction in a manner similar to the fuzzy "AND" operation to quantify the agreement among themselves. To implement this fuzzy "AND" operation given four vortex detectors, we apply a majority vote algorithm as :

$$f(x_1,x_2,x_3,x_4) = \begin{cases} 1, & \text{if } \sum_{i=1}^{4} I(x_i,t_{high}) \geq 3 \\ 0, & \text{if } \sum_{i=1}^{4} I(t_{low},x_i) \geq 3 \\ -1, & \text{Otherwise} \end{cases} . \tag{5.10}$$

Here, t_{high} and t_{low} are two threshold values such that $0.0 < t_{low} < t_{high} < 1.0$ and $I(\cdot)$ is a boolean comparison function which is defined as:

$$I(a,b) = \begin{cases} 1, & \text{if } a \geq b \\ 0, & \text{if } a < b \end{cases}. \tag{5.11}$$

Here t_{high} denotes the threshold above which the points can be marked as vortex with a high degree of certainty. Similarly, t_{low} denotes the threshold value below which the points can be marked as non-vortex with higher confidence. The values falling between these two thresholds are considered uncertain points that need further attention. Thus, Equation 5.10 defines a majority voting scheme based on these two parameters. For a given point, if at least three of the four detectors assign a vortexness value that is at least t_{high}, then this point is marked with 1. Similarly, if three out of four detectors provide a value less than t_{low}, then this point is initially marked with 0. The rest of the uncertain points are assigned the label -1 and these points are then further processed according to their proximity from the already detected certain vortex points as discussed next in Section 5.2.3. The estimation of the thresholds t_{high} and t_{low} is discussed in Section 5.2.5.

5.2.3 Uncertainty Reduction Through Spatial Proximity

Analysis of individual data points alone, to determine their likelihood of being part of a vortex, limits the effectiveness of the detection process. We propose that the incorporation of spatial locality into the pointwise vortex detectors may enhance the capabilities of the detector by making it less localized. If it is certain that one point is part of a vortical flow, then the likelihood of its neighbors being in a vortical flow also increases. On the contrary, if it is known that a point is surely a non-vortex, then its neighbors also become more likely to be non-vortex. This spatial neighborhood feature can be incorporated into our system as follows. After identifying the more certain candidate vortex regions with the oracles

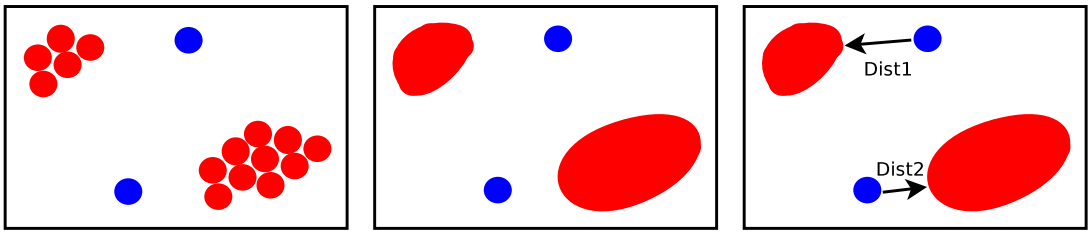


Figure 5.2: Incorporation of spatial proximity. (a) Certain (red) and uncertain (blue) points detected after majority voting. (b) Spatial clustering of certain points. (c) Uncertain points are classified according to their distance from their nearest certain vortex cluster.

as shown in Section 5.2.2, we first perform spatial clustering of these points. Thus, we have in effect identified salient points near the vortex core. The clustering is performed using a "region fill" approach in which each point labeled as a vortex starts as an individual cluster, which is then merged with the neighboring clusters and, in turn, grown into even larger clusters. Next, we analyze the remaining uncertain points based on these clusters. After the identification of the clusters, we take advantage of the spatial locality for the uncertain points that have the label -1. For each of these points, we determine the nearest vortex cluster and calculate the distance d from that cluster. The vortexness values of these uncertain points are readjusted according to a distance based decay function, D.

$$D(d) \propto \frac{1}{d^2} \Leftrightarrow D(d) = \frac{K}{d^2}. \tag{5.12}$$

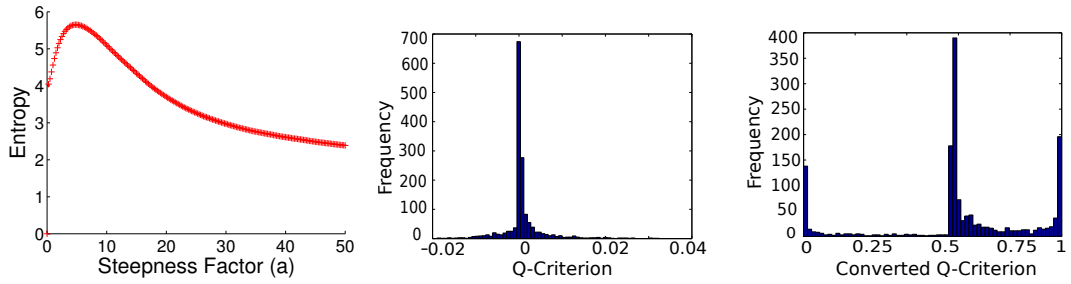
Decay is modeled as an inverse square function. The proportionality constant K is dependent on the data and is estimated using domain expert's input for higher accuracy as discussed in Section 5.2.5. Suppose, x_{avg} denotes the average vortexness value of an uncertain point calculated from the four vortex detectors. Now, for all the uncertain points, x_{avg}

is adjusted based on distance and these uncertain points are finally classified according to the previous high-threshold t_{high}.

$$f(d, x_{avg}) = \begin{cases} 1, & \text{if } x_{avg} * (1 + \frac{K}{d^2}) \geq t_{high} \\ 0, & \text{Otherwise} \end{cases}. \tag{5.13}$$

Figure 5.2 illustrates this stage of the framework. The individual certain vortex points (marked in red) of Figure 5.2(a) are spatially clustered in Figure 5.2(b). The uncertain points (marked in blue) are then recalculated for their vortexness according to their distance from the nearest cluster as shown in Figure 5.2(c).

5.2.4 Parameter Selection Using Information Theory



(a) Change of entropy with the change in steepness parameter a.

(b) Original distribution of Q-criterion before applying entropy maximization.

(c) Distribution of Q-criterion after applying entropy maximization.

Figure 5.3: Change in the distribution due to entropy maximization.

The fuzzy conversion function $F(x)$ consists of two parameters, th and a, as shown in Equation 5.8. Since each individual vortex detector is assigned a function $F(x)$ to convert to the fuzzy output, the parameters of $F(x)$ also depend on the properties of the individual detector. Since the output of $F(x)$ is most uncertain when $x = th$, the value of th is set to the

theoretical threshold value of that vortex detector which is known for each detector as described earlier. This describes the property that the pointwise detectors are most uncertain around their theoretical threshold.

The steepness parameter a has two components: the sign and the magnitude. Since the negative values for some of the detectors are more likely to be a vortex (e.g., λ_2) whereas for others, the positive values (e.g., Q-criterion) represent vortex regions, the sign of a is chosen to make all the detectors consistent after fuzzy conversion. The sign is determined by using the function S as follows:

$$S(a) = \begin{cases} +, & \text{if } L(x) = 1,\ \forall x > th \\ -, & \text{if } L(x) = 0,\ \forall x > th \end{cases} \tag{5.14}$$

where for a given vortex detector, L is the previously defined labeling function, th is corresponding theoretical threshold and $x \in \mathbb{R}$.

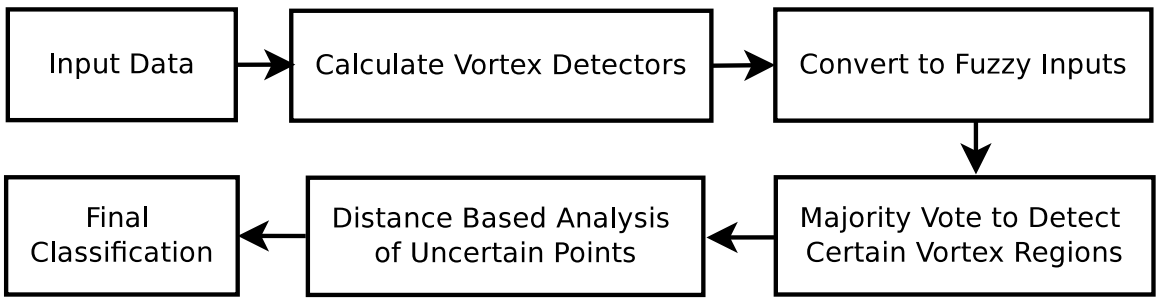
After determining the sign, the magnitude of a must be determined. If $|a| \approx 0$, then after the fuzzy conversion of the vortex detector, the histogram of the resulting vortexness values shows a very high peak near 0.5. This reflects that, after fuzzy conversion, all the values of the vortex detector have mapped to the uncertain 0.5 on the fuzzy scale. Similarly, if $|a|$ is very large, the resulting histogram of vortexness values reduces to two peaks, 0 (certain non-vortex) and 1 (certain vortex), representing the hard-thresholding scheme. We strive for a value of the steepness factor a that generates a vortexness value distribution that is between these two extremes such that the converted fuzzy vortexness values are well separated. This conversion will represent the confidence of the vortex detector and to achieve a well-distributed histogram after fuzzy conversion, we apply the notion of entropy maximization [10, 31]. For a random variable X, if $p(x)$ denotes the probability of occurrence

of an observation $x \in X$, then entropy $H(X)$ is defined as

$$H(X) = -\sum_{x \in X} p(x) \log p(x). \tag{5.15}$$

An increase in the entropy value for a random variable generally signifies that the values of the variable are well distributed in the different bins of its histogram. In our entropy maximization method, the parameter a is chosen such that it maximizes the entropy of the distribution of resulting fuzzy vortexness values of the detector after the application of the sigmoid function. Figure 5.3 shows results from rearward facing step dataset [3], which is described in more detail in Section 5.3.2. In this example, the application of entropy maximization changes the unimodal distribution of detector values into a distribution with three peaks. Instead of simply applying entropy maximization, we use the parameter value that maximizes the product of entropy and the rate of change of entropy to reduce the points mapped to the values 0 and 1. This enhances the separation of data points based on the transformed values in the following stages and thus increasing the efficiency of the algorithm.

5.2.5 System Integration with Domain Expert's Input



(a)

Figure 5.4: A schematic view of the different stages of our system.

In this section, we integrate the different components of our system as shown in the schematic diagram of Figure 5.4 and elaborate the role of the domain expert. We worked closely with a domain expert who employed a global streamline method to generate the labels for points in the field which will be used to quantitatively measure and compare our proposed method with the existing ones. Our approach is similar to the one described in [153] that is based on Robinson's definition of a vortex [107] and the observation that a Galilean transformation to an appropriate reference frame is necessary for unsteady flows with moving vortices [141]. The domain expert uses a manual vortex detection process whereby a set of streamlines is generated from seed points placed by the expert. Since streamlines are not Galilean invariant, the domain expert must adjust the local reference frame [141] in an iterative process to make sure that the swirling nature of the streamlines is accurately captured. The goal of the expert is to select a set of parameters that produce a coherent set of streamlines. To aid in this process, the domain expert initially uses a point based detector, such as the λ_2 method to generate isosurfaces that identify the candidate vortical regions in the field. The domain expert then employs a "point picking interface" by which he can select regularly-shaped regions in the field to explore further with streamlines. If the domain expert observes a coherent set of swirling streamlines, then he changes the view point such that the vortex axis is aligned with the view plane normal. Then these spatial three-dimensional points are marked with 1 for vortex and 0 for non-vortex. Further refinement can be obtained by picking individual points and changing their labels.

With the expert labels available, we now integrate all the different components of our system in this section. For a given dataset, the four vortex detectors- Q-criterion, λ_2 method, Δ criterion, and Γ_2- are first calculated. These outputs are converted to fuzzy values by using the sigmoid functions that maximize the product of entropy and rate of

change of entropy of their histograms as described in Sections 5.2.1 and 5.2.4. Then majority voting is applied based on the two thresholds: t_{high} and t_{low}. The low threshold t_{low} is set to 0.25 for all the experiments. For increased accuracy, the high threshold t_{high} is estimated from a small fraction of the domain expert's labels. At this stage, a small number of samples (typically 10%) are randomly chosen and t_{high} is selected such that the false positives are minimized given these random samples. The data points are now marked with certain 0 or 1 labels and uncertain -1 labels. In the next stage of our pipeline, the certain 1 values are spatially clustered. Then all of the uncertain -1 data points are separated and their distances from the nearest vortex cluster are computed. Based on these nearest distances, the uncertain points are classified. The proportionality constant K in Equation 5.12 is also estimated from the same set of samples drawn from the domain expert's labels such that the final error rate and the true positive rates are higher than each individual detector's performance. Finally, the remainder of the domain expert's labels is used as a basis of comparison for generating results to quantitatively assess the effectiveness of the different vortex detection methods.

5.3 Experimental Results

The experiments were conducted on a Linux machine with 2.40 GHz Intel core i7 CPU, 8 GB of RAM and an NVIDIA Geforce GT 650M GPU with 2GB texture memory. For generating comparative results, we use the labels generated by the domain expert as a basis of comparison for the below mentioned datasets. The following metrics are used for

quantitative comparison among different vortex detection techniques:

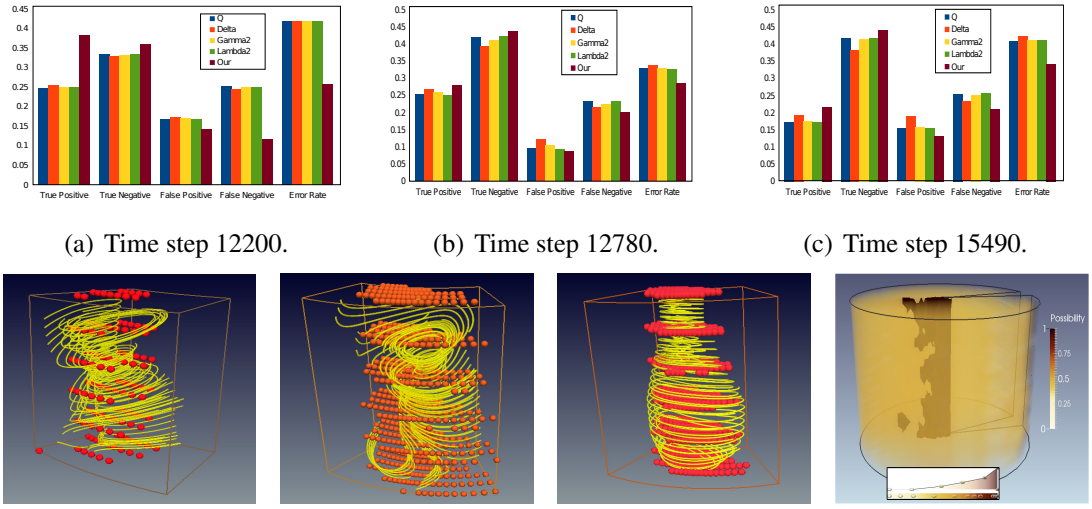
$$
\begin{aligned}
ErrorRate &= (P_F + N_F)/(T+N) \\
TruePositiveRate &= P_T/(T+N) \\
TrueNegativeRate &= N_T/(T+N) \\
FalsePositiveRate &= P_F/(T+N) \\
FalseNegativeRate &= N_F/(T+N)
\end{aligned} \tag{5.16}
$$

where P_T denotes the true positive count, N_T denotes the true negative count, P_F is the false positive count, N_F is the false negative count, T is the total vortex count in expert labels and N is the total non-vortex points in expert labels. The parameters of our system, high threshold t_{high} and distance based constant K, are tuned using random samples (10%) from the expert labels with the performance of the algorithms measured on the rest of the labeled points.

5.3.1 Tapered Cylinder Data

Our method is first applied to the Tapered Cylinder dataset [35], which describes an unsteady, three-dimensional, incompressible, laminar, viscous flow around a cylinder with its axis oriented perpendicular to the primary flow. It should be noted that, since the cylinder extends across the entire domain, there are no tip effects present in the simulation.

We compare our proposed method with the four existing vortex detection algorithms used as oracles for our system : Q-criterion, λ_2 method, Δ criterion, and Γ_2 method. Figure 5.5(a) shows comparative results for the time step 12200 of this dataset with Y-axis showing the percentage count of the data points. It is readily visible that, for this time step, our proposed method significantly increases the true positive rate and reduces the total error



(a) Time step 12200. (b) Time step 12780. (c) Time step 15490.

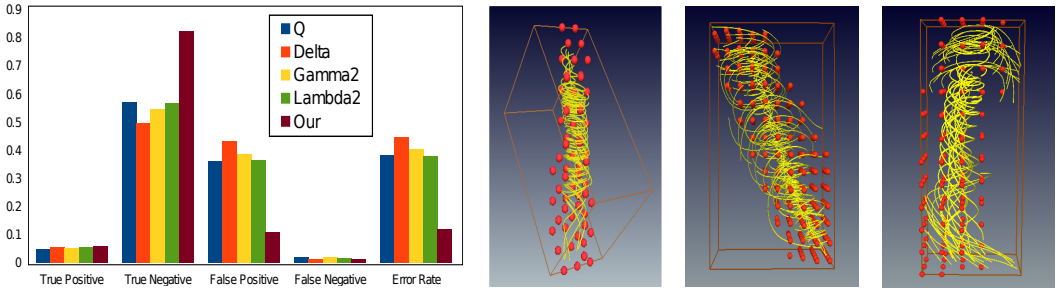
(d) Expert labels for 12200. (e) Expert labels for 12780 . (f) Expert labels for 15490. (g) Visualization of our prediction field.

Figure 5.5: Results for different time steps of the Tapered Cylinder dataset.

rate. It is also observed that the true negative rate is better than any of the four existing detectors and false positive and false negative rates are consequently reduced. Figure 5.5(d) represents an example where the domain expert used the global streamline method to generate the labels for this time step. Here, the red spheres indicate the expert's vortex labels and streamlines were seeded in the region after adjusting the reference frame. For more details about the streamlines and selection of reference frame, we refer the interested readers to the supplemental material. Our improved error rate confirms that our method is more consistent with the domain expert's labelings. Experiments were further conducted on other time steps, which are far apart in the temporal domain. Figure 5.5(b) and Figure 5.5(c) show the results obtained from 12780 and 15490 respectively. In each of these time steps, our proposed method provides improved results for all types of errors. Examples of the domain expert's labelings are shown in Figure 5.5(e) and 5.5(f) for time steps 12780 and

15490 respectively. The results show a good agreement with the domain expert's markings which in turn shows more robust result generated by our method. The false positive rates of the points that were decided by spatial proximity criteria are 0.09, 0.05 and 0.11 respectively for the time steps 12200, 12780 and 15490. In Figure 5.5(g), the volume rendering of the final prediction field generated from our algorithm for the time step 12200 is shown. Here the vortex regions of higher confidence are suitably highlighted by setting the transfer function.

5.3.2 Rearward Facing Step



(a) Time step 12200. (b) (c) (d)

Figure 5.6: Results for the Rearward Facing Step dataset. (a) Quantitative comparison results. (b)-(d) Domain expert's labeled points in three regions of the dataset.

We now apply our method to a dataset that models the unsteady, incompressible, turbulent flow over a rearward facing step. The flow enters the domain from the lower left in the positive x direction, encounters the step, and is separated. The region of separated flow is highly turbulent and contains vortices. In this case, the axis of a given vortex may have significant curvature and its orientation with respect to the primary flow may vary widely from other nearby vortices. The flow conditions were chosen to match the experimental

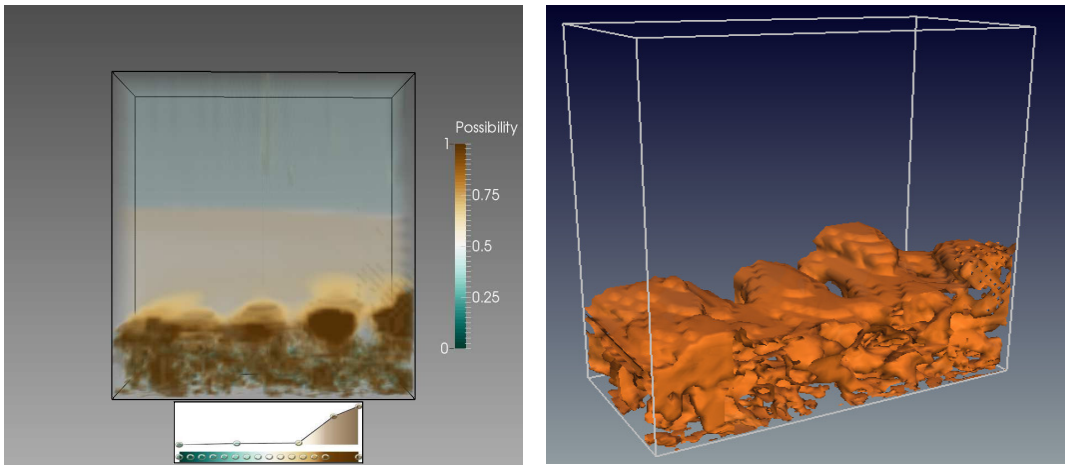
data obtained by Driver and Seegmiller[37]. Details concerning the numerical simulation are reported in Alam *et al.* [3].

Figure 5.6(a) shows a comparison of the four vortex detectors and our proposed method to provide a complete quantitative analysis. As can be seen from this plot, the overall true positive rate is slightly better than the existing detectors, since the total number of vortex points is smaller in this dataset, but the major performance gain is achieved in the true negative rate. Our proposed method was able to eliminate most of the false positive labels and thereby significantly reduce the total error rate. Figure 5.6(b), Figure 5.6(c) and Figure 5.6(d) show three regions in the dataset where the domain expert extracted the voritcal motion by adjusting the correct reference frame (more details provided in the supplemental material). Here, the red spheres indicate the expert's vortex labels. Figure 5.6(a) shows that our results conform to the domain expert's labels and was able to identify the vortices with improved accuracy. For this dataset, the false positive rate of the points that were classified by the spatial proximity criteria was 0.12, which is close to the overall false positive rate as shown in Figure 5.6(a). Figure 5.7(a) shows the volume rendering of the resulting possibility field of a region of this dataset with the transfer function shown below that emphasizes the vortex regions of higher confidence. We note that the bottom part of the dataset is more turbulent. Finally, we show the isosurface generated from our prediction field in Figure 5.7(b) depicting the detected vortex regions.

5.4 Discussion

5.4.1 Comparison of Algorithms

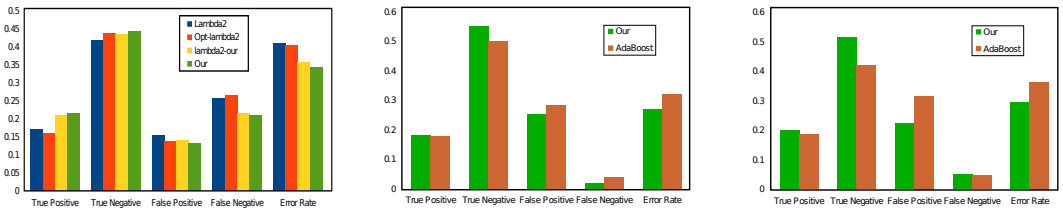
In this section, we compare the results of our proposed framework with other techniques that benefit from the labels generated by the domain expert. In Figure 5.8(a), the four



(a) Time step 12200. (b)

Figure 5.7: Results for the Rearward Facing Step dataset. (a) Volume rendering from our algorithm generated prediction field. (b) Isosurface from the prediction field showing the vortical regions detected from our algorithm.

different scenarios are presented from time step 15490 of the Tapered Cylinder dataset. In this figure, one set of results is taken after generating the optimized hard-threshold [153] of an existing detector, namely the λ_2 method as it is widely used given its reliability compared to the other pointwise metrics. "Opt-Lambda2" depicts this optimized threshold selection in the figure with "Lambda2" as the theoretical hard-thresholded λ_2 example. Next, we modify our proposed algorithm and, instead of using a consensus of four existing detectors, we use λ_2 as the only input to our system, keeping all the other stages of our proposed method the same. This situation is shown as "lambda2-our" in Figure 5.8(a). For comparison purposes, our complete proposed algorithm is also shown in the figure and is labeled as "Our". The results show that the use of an optimized threshold improves the error rate but at the expense of a reduced true positive rate. Both "lambda2-our" and "Our" outperform "Lambda2" and "Opt-lambda2", which suggests that the use of spatial locality



(a) Time step 12200. (b) (c)

Figure 5.8: Comparison results for the Tapered Cylinder dataset. (a) Comparison with different modifications of our proposed algorithm from time step 15490. (b) Comparison with AdaBoost from time step 12210. (c) Comparison with AdaBoost from time step 12240.

information is an important factor for effective detection of vortices. The comparison of "Our" and "lambda2-our" shows that consensus from four oracles is better than trusting just one, especially when all of the oracles are fallible and the best oracle for a given dataset is not known a priori. method as it is widely used for its reliability compared to the others pointwise metrics. The exclusive use of any of the other three existing metrics instead of λ_2 produces similar results.

Next, we compare our method with the AdaBoost method, which was recently proposed by Zhang et al. [153]. This AdaBoost method also uses multiple vortex detectors as its input and tries to optimize the performance using samples from the domain expert. To conduct the experiment, we selected the previously mentioned 12210 time step from the tapered cylinder dataset. In the Figure 5.8(b), "Our" denotes the method proposed in this chapter and "AdaBoost" is the method proposed in [153]. To estimate the system parameters for both the methods, 10% of the domain expert's markings from the data were used and the results were generated. Also, to test the reliability and sensitivity of the two methods, we conducted another set of experiments in which we used 10% of the domain

expert's markings selected only from time step 12210 to tune the system parameters and applied the estimated parameters to the later time step 12240. Here we assume that the time-varying flow field properties will be changing slowly over the time interval and the parameters estimated from the previous step should be applicable to the next with only a slightly increased error rate. The results for these are indicated by "Our" and "AdaBoost" in Figure 5.8(c). From Figure 5.8(b) and 5.8(c), it can be observed that even if the parameters are tuned at each time step or if the parameters are estimated from an initial time step and re-applied to a later time step, our method outperforms the AdaBoost method.

5.4.2 Performance

We now discuss the performance of different components in our proposed framework as shown in Table 5.1. Here, TC refers to Tapered Cylinder dataset and RFS denotes the Rearward Facing Step dataset. In the first column, we present the time taken to generate the four vortex detection methods from the curvilinear grid datasets. The time taken at this stage depends on the data size and complexity of the gradient computation. The next stage denotes the mutual information maximization stage that is used to convert the four vortex oracles to fuzzy input. We used the *fminsearch* function provided by Matlab at this stage. The next column shows the majority voting scheme which polls the four oracles to decide the certain region. The last column presents the time for distance computation of the uncertain points from the more certain vortex regions. This stage uses the NVIDIA Thrust library to exploit GPUs for efficient distance computation. The computation time of this stage depends on the data size and the number of uncertain points together with the more certain clusters. Using *fminsearch*, this stage requires four to five iterations for our case.

Table 5.1: Running time for different components of the framework.

	Vortex Data Generation (Sec.)	Mutual Information Maximization (Sec.)	Majority Vote Scheme (Sec.)	Cluster Distance Computation (Sec.)
TC	33.8	4.5	0.007	24.5
RFS	61.3	8.3	0.021	71.7

5.5 Conclusions

In this chapter, we proposed a novel uncertainty-based vortex analysis framework. This approach examines the uncertainty contained in four existing local vortex detection methods and combines them to perform the vortex analysis and detection in a more robust way. Since there are multiple vortex detectors available, these detectors are used as the inputs to the system and the uncertainty of their predictions is modeled using a sigmoid function. This sigmoid function converts the outputs of these detectors to a possibility value in the range of 0 and 1 where these possibility values denote the certainty that a point will be classified as a part of a vortex region. A voting algorithm is applied to classify the more certain vortex regions and cluster these regions based on their spatial locations. Next, we introduce the use of spatial proximity as an uncertainty reduction stage and classify the remaining points as vortex or non-vortex based on their distance from the closest vortex cluster. We worked closely with a domain expert and used the data marked by the expert to compare our method with other existing methods. We applied our method on multiple datasets and time steps and showed that the accuracy can be improved by systematically analyzing these flow structures.

Chapter 6: An Error-Aware Flux-Based Algorithm for Scalable Streamline Generation

In the previous chapters, we discussed the uncertain existence of relationships and features in datasets. Now we turn our focus towards the error and uncertainty analysis of flow visualization tools, specifically streamlines in this chapter and stream surfaces in the next one. These tools are extensively used for visualization of flow fields. A streamline is traced out from a given seed point and integrated over the flow field so that it is everywhere tangent to the flow field. Streamlines are intuitive, simple to calculate, and they reveal the flow characteristics quite well. The need to analyze large flow datasets has driven the researchers to strive for strategies that can generate streamlines in parallel with minimal error. Traditionally, parallel streamlines are generated by parallelizing over the vector field domain and/or over the seeds, where the path of a streamline is computed serially by one processing element (PE) at a time (which may be passed from PE to PE). Generating a single streamline in parallel by multiple processing elements has largely remained unsolved due to the dependency among the consecutive streamline segments. A parallel algorithm that can generate individual streamline segments simultaneously, will reduce the process of particle advection to only local computations and make it highly parallel. In this chapter, we present such a new parallel method to construct a streamline in parallel segments

using a flux-based technique for large two-dimensional flows. Analysis of large two-dimensional flow fields is of prime importance and often the most preferred means in many research areas and applications, including ocean and climate modeling that explore large three-dimensional datasets slice-by-slice. Our proposed algorithm addresses the needs of these scientific domains by exhibiting much-improved execution time, scalability and load balancing when compared to the traditional parallel advection methods.

Apart from the correctness, another reason for developing a new parallel streamline computation method is for scalability and to match the computational properties of the source simulations. Flow simulation models such as parallel ocean models are frequently computed using a finite-element (FE) or finite-volume (FV) method that scales well in parallel, by dividing the domain among the processing elements and they achieve nearly-perfect load balancing. With an FE/FV method, all processing elements have an equal amount of work through independent computation on several domains and they bulk-synchronize through "ghost cell" communication. The various known parallel streamline methods do not follow this computational and communication pattern, as they use a mixture of strategies to achieve run-time load balancing. Frequently, they shuffle the domain or seeds among processors. In contrast, our new parallel streamline algorithm matches the computational, communication, and load-balancing properties of the large flow simulation models (e.g., Parallel Ocean Program from Los Alamos National Laboratory), by parallelizing over the domain, and it is able to utilize and run in the same parallel framework as the source simulation. We achieve this through a parallel flux computation and isosurfacing strategy.

Instead of computing streamlines as a series of integrations, an alternative solution is to calculate implicit streamlines by deriving a flux-based scalar field from a given vector

field. Then the isocontours of this field represent the streamlines. Computation of flux-based isocontours to represent streamlines can be achieved efficiently in two dimensions and for incompressible flows. Since the generation of isocontours in parallel is well optimized, in this chapter we take up this approach to generate streamlines for large-scale two-dimensional flows. Apart from being load balanced, this method provides faster data exploration through the use of a large number of streamlines. Also, space usage is reduced since the original vector field can be discarded benefiting the storage and processing of large datasets.

In our method, we adopt a novel divide and conquer strategy to analyze large two-dimensional datasets. We formulate a parallel streamline generation method that uses the load balanced parallel data computation model of the existing flow simulations. Given a dataset, initially it is divided into multiple data blocks and flux-based scalar fields are computed in these blocks in parallel. The isocontours of these local flux-fields represent the streamlines in those local blocks. The local data is then globally synchronized by propagating the flux offset values that allows for stitching of the shorter streamline segments. Finally, the complete streamlines are generated in parallel by calculating isocontours of the data blocks in parallel. Although this method is currently designed to work in the multi-processor environment, any GPU-based architecture can also benefit from this algorithm.

Our contributions in this chapter are multi-fold:

1. For analyzing large-scale two-dimensional flow simulation data, we use a flux-based approach to compute the local streamlines of a region and assign flux ids to the streamlines to store them in the form of a scalar field.

2. We propose a novel algorithm to stitch the different segments of a streamline seamlessly across local regions to produce a longer streamline.

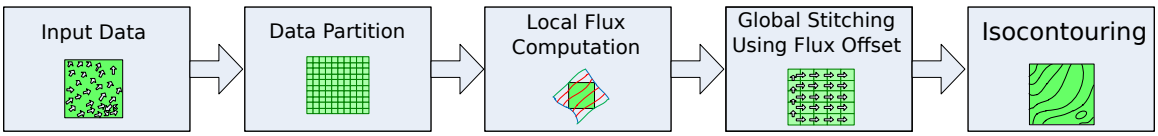


Figure 6.1: A schematic representation of our system pipeline.

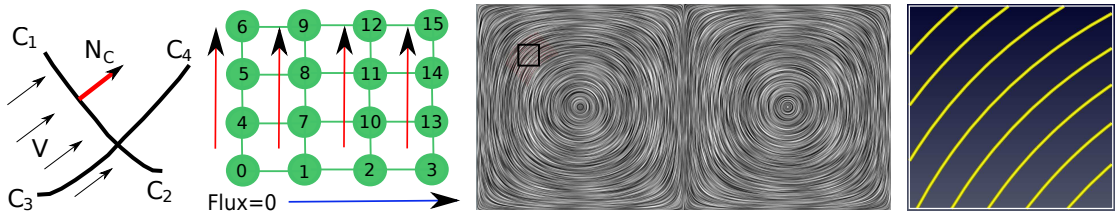
3. Our streamline generation algorithm provides an efficient multi-processor workflow to achieve speed up, scalability and load balancing.

6.1 System Overview

In this section, we briefly discuss the different stages of our pipeline. The goal of this chapter is to generate streamlines in a distributed manner. We want to generate the different segments of a given streamline independently and stitch them seamlessly over the different blocks. To achieve this, we leverage the flux invariance property of incompressible flows. Given the dataset, first the dataset is subdivided into different blocks. After this data partition stage, a local computation is applied on each individual block. In this local computation stage, the vector data is converted to a scalar field such that the isocontours of the resulting scalar field represent the streamlines. To achieve this, the flux values of the local block are computed to implicitly store the local streamlines. After the vector field of local blocks are converted to scalar flux field, the flux values are synchronized across the blocks. To achieve this, we exploit the additive property of the flux values and each block sends out its flux offset value to its neighbor. By propagating the flux values across the neighbors, the global flux field is synchronized. To generate the final streamlines, each block can extract the isocontours independently which are stitched across

the blocks seamlessly. These isocontours represent streamlines for the input vector dataset. A schematic view of our workflow is presented in Figure 6.1.

6.2 Methodology



(a) Two-dimensional flux across planar curves.

(b) Local flux computation process.

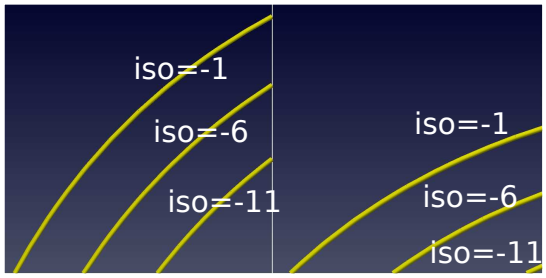
(c) LIC visualization of the Double Gyre dataset.

(d) Isocontours from the flux field.

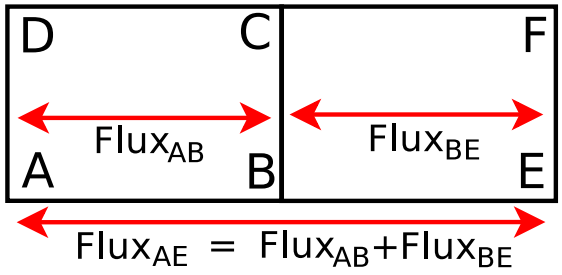
Figure 6.2: An illustrative example of flux-based streamline computation.

6.2.1 Flux-based Ordering of Streamlines

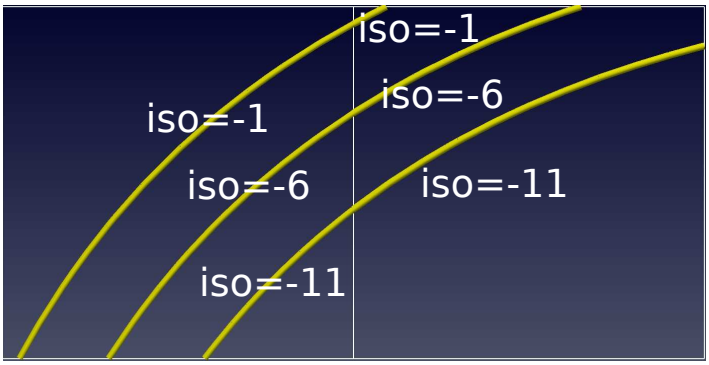
Given a large two-dimensional vector field, $\vec{V} =< u, v > \in R^2$, initially the dataset is decomposed into several blocks to initiate the parallel local streamline generation process. In the local computation stage, the goal is to assign unique local identifiers to the streamlines of each data block without explicitly computing them. Since streamlines do not intersect each other except at the critical points, each point in the spatial domain can be assigned a scalar as the streamline identifier, where points with the same scalar are part of the same streamline. This transformation from the input vector field to a scalar field can effectively be used to implicitly compute the streamlines. Although there are many possible ways of finding unique streamline identifiers, assigning the identifiers based on flux



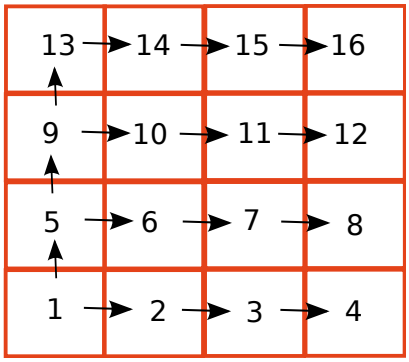
(a) Isocontours generated in parallel without stitching.



(b) A schematic example of the relationship of the flux values for the neighboring data blocks.



(c) Isocontours generated in parallel after stitching across the blocks.



(d) A simple flux propagation scheme among the data blocks.

Figure 6.3: Flux based stitching and parallel propagation.

values is more advantageous because, a) no explicit streamline computation is necessary, and b) these local streamlines can be stitched across the blocks efficiently in the later stages of our pipeline according to the flux conservation property of the flow. Compared to this flux-based method of streamline ordering, other methods such as employed by Van Wijk [135], do not allow for efficient and smooth streamline stitching as they are not based on the physical property of the fluid.

In the field of transport phenomena, flux refers to flow per unit area. For a given quantity, this is measured as the amount of transport normal to the infinitesimal surface patch in three-dimensions. In two-dimensional scenario, flux Φ across a two-dimensional curve is

defined by the amount of flow that is normal to that curve:

$$\Phi(\vec{C}) = \int_{\vec{C}} \vec{V} \cdot d\vec{c} = \int_{\vec{C}} \vec{V} \cdot (\vec{N}_c * dc) = \int_{\vec{C}} (\vec{V} \cdot \vec{N}_c) dc \tag{6.1}$$

where the integral is performed along the curve $\vec{C}$ and $\vec{N}_c$ is the unit normal to the segment $d\vec{c}$. A schematic example is shown in Figure 6.2(a) where curve C_1C_2 with normal N_C is placed in the vector field V. In this vector field, if C_3C_4 is a streamline then the flux across this curve is 0. The key property we leverage here is that flux is conserved within a control area for steady state incompressible fluid flows with zero divergence, i.e., divergence $div(\vec{V})$ at every point follows this:

$$div(\vec{V}) \equiv \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0. \tag{6.2}$$

So, the total flow going into a finite area will be equal to the total flow going out of that area.

As a discrete approximation of Equation 6.1, total flux $Flux_{A_1A_n}$ across a piecewise linear curve A_1A_n where n points on the curve $A_1A_n = \{A_i\}, i = 1...n$, can be computed as :

$$Flux_{A_1A_n} = \sum_{i=1}^{n-1} (((V(A_i) + V(A_{i+1}))/2) \cdot N(A_iA_{i+1})) * d(A_iA_{i+1}) \tag{6.3}$$

Here, $V(A_i)$ returns the vector of location A_i, $N(A_iA_j)$ denotes the unit normal to the segment A_iA_j whose length is given by $d(A_iA_j)$ and $Flux_{A_iA_j}$ denotes the flux of segment A_iA_j with A_i as the flux origin point. Here we note that, $Flux_{A_iA_j} = 0$ if $i = j$. Instead of the trapezoidal rule presented in Equation 6.3, any other method (e.g. Simpson's rule) can also be applied to evaluate the integral of Equation 6.1.

To generate a field of streamline identifiers, we compute the flux values from a flux origin for every grid point. Assuming an arbitrary flux origin point A_1 within a data block, flux computation can be performed for a grid point A_n by applying Equation 6.3 along a

path A_1A_n that connects the grid point and the local flux origin. Extending this to all the grid points of the block, the local vector field can be converted to the intended flux field. In our case, as shown in Figure 6.2(b), we take the left-bottom most point of the rectangular block as the local flux origin and propagate the flux values to the neighboring grid points until all the grid points of the block are assigned flux values given the local flux origin. First, the flux propagation is performed along the lower edge (along the blue arrow) and then the flux values are propagated column-wise (along the red arrows). The time stamps shown in Figure 6.2(b) represent the order in which the grid points receive the propagated flux values. The isocontours of thus produced flux field yield the streamlines. In Figure 6.2(c), the Double Gyre dataset is visualized using the popular Line Integral Convolution (LIC) technique and a 10×10 block is extracted as shown by the black box. The yellow curves of the Figure 6.2(d) are the isocontours of the flux field that show a good conformance with the numerical integration based streamlines from the vector field. Next, we discuss how this flux-based ordering facilitates effective streamline stitching across blocks.

6.2.2 Flux-based Stitching of Local Blocks

Using the grid based flux computation approach, it is possible to generate local isocontours as local streamline segments in parallel that are coherent within individual data blocks. But when two neighboring blocks generate two segments of one streamline independently by generating isocontours of a given flux iso-value, the two isocontours (i.e., the two streamline segments) across blocks may not match at the block boundary as shown in Figure 6.3(a). This is because the flux values at each block were computed assuming local flux origins. Figure 6.3(a) shows three isocontours from two neighboring blocks where they show a mismatch at the block boundary due to local computation. To achieve a proper

stitching of the streamline segments across blocks, communication among the blocks is needed to synchronize all the blocks to a common flux origin.

When an incompressible fluid flow is decomposed into blocks, two neighboring blocks produce flux values assuming local flux origins. Since flux is conserved across the blocks and is independent of the path, for two neighboring blocks to have consistent flux values, their flux origins need to be aligned. This synchronization of flux values can be achieved using the boundary points shared by the neighboring blocks. Following Figure 6.3(b), two neighboring blocks ABCD and BEFC initially compute local flux values with respect to flux origin points A and B respectively. Due to this mismatch of flux origins in the two blocks, given an iso-value, the local isocontour segments of the individual blocks are not globally coherent. Evidently, in general scenario, $Flux_{AB} \neq Flux_{BB}$ which causes the mismatch and the flux values of the block BEFC require a change of origin to be aligned with global flux origin A instead of local origin B. The key property we leverage here is that when flux is conserved in the domain, the flux difference between two points B and C is independent of flux origin. In other words, when flux difference between B and C is computed assuming flux origin A within block ABCD as $\Delta F_1 = Flux_{AC} - Flux_{AB}$ and the same flux difference is computed assuming flux origin B within block BEFC as $\Delta F_2 = Flux_{BC} - Flux_{BB}$, then :

$$\Delta F_1 = \Delta F_2 \Rightarrow Flux_{AC} - Flux_{AB} = Flux_{BC} - Flux_{BB}, \tag{6.4}$$

Following this, synchronization of flux values across the blocks is achieved by observing the flux difference between the local and global origins, $Flux_{AB} - Flux_{BB}$ and adding this flux offset to all the flux values of block BEFC. Thus, flux values can be directly communicated whenever two blocks share a common grid point or a common edge. In general, if two blocks $B1$ and $B2$ share a common point i with local flux origins p_1 and p_2 respectively,

then change of flux origin from p_2 to p_1 for $B2$ is achieved as:

$$Flux_{p_1 j} = Flux_{p_2 j} + (Flux_{p_1 i} - Flux_{p_2 i}), \forall j \in B2. \tag{6.5}$$

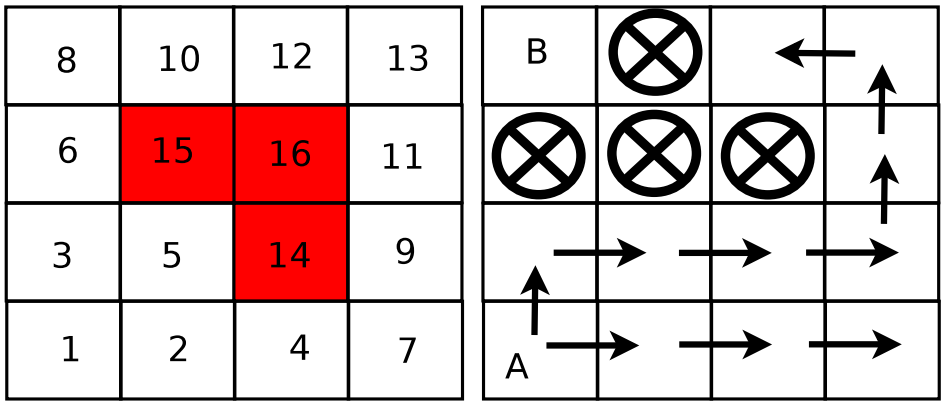
If $B1$ and $B2$ share a common edge, then any point i from that shared edge can be chosen for this change of flux origin following the flux conservation theory. To ensure that the neighboring blocks have a shared edge or a point, we use one layer of ghost cells in the data decomposition phase.

Given multiple data blocks and locally computed flux values, a parallel communication algorithm can be devised to facilitate the flux offset propagation as shown in Figure 6.3(d). We assume that the global origin is contained in the left-bottom most block. Thus, in Figure 6.3(d), block 1 contains the global origin and it sends the required flux offset value to block 2 and block 5 in the first phase of communication. Next, after receiving the offsets, the two blocks adjust their local flux values to align with block 1 and then further send out the flux offsets to their neighbors, block 3, block 6, and block 9. The overall communication takes $O(N^{\frac{1}{2}})$ stages to complete where N is total number of data blocks. Since only a single flux-offset value is to be sent among neighboring blocks, the communication cost is much less compared to seed or data exchange in traditional parallelize-over-block or parallelize-over-seed techniques. After this communication stage, the resulting stitched flux field can be used to generate isocontours that represent streamlines. Since isocontour extraction routines are highly parallel and optimized for performance, the final streamline generation stage is also very efficient. After this stitching, as shown in Figure 6.3(c) compared to 6.3(a), now isocontours of the same iso-value can be generated independently in two neighboring blocks that match seamlessly the block boundary.

6.2.3 Error Handling

Handling Non-zero Divergence Regions

The flux-based parallel streamline generation and stitching algorithm provided above works very well for two-dimensional datasets that have zero divergence. Although incompressibility is a popular assumption in flow simulations, the real datasets can consist of regions that contain non-zero divergence values and the local flux values at these regions contain uncertainty. Since flux computation is additive, this uncertainty can spread from one region to another if a simple flux computation algorithm is employed, e.g., the method shown in Figure 6.2(b). To restrict this uncertainty from spreading throughout the domain, we devise an error-aware flux computation in the local blocks. Here, we use divergence of the field as the error indicator as described by the Equation 6.2 and we assume an additive error model since flux is also additive. Given a data block, a conceptual graph is created where each node represents a grid point and absolute values of divergence errors are assigned to the nodes as their weights. Undirected edges are present if two nodes are neighbors. Now, a possible flux computation path is found by finding a path starting from the start node that visits all the nodes where the node weights are propagated at each hop. Our goal is to find a path that minimizes the sum of flux errors of all the nodes of this graph after the flux values are computed for all nodes. Instead of finding all possible spanning trees of the graph, we apply a greedy flood fill like algorithm similar to Dijkstra's algorithm for the single-source-shortest-paths problem. The source node is taken as the node with minimum divergence error and we maintain a sorted list that determines the next node to be visited. The neighbors of the currently explored node are put into this list sorted according to their divergence error values. This greedy scheme generates good quality results with a small performance overhead.



(a) A schematic example of handling regions with non-zero divergence values. Red regions indicate larger divergence regions.

(b) A schematic example of handling regions with discontinuities. The marked-circular glyphs indicate cluster of discontinuities.

Figure 6.4: Handling of erroneous regions while propagating flux.

A schematic example of this is provided in Figure 6.4(a) where a 4×4 block is shown and red marked regions are non-zero divergence regions. Instead of applying a naive algorithm, our error-aware algorithm defers the exploration of the erroneous regions which in turn restricts the error propagation. The numbers show the timestamps of regions as they were explored. Global origin point is now selected to be the grid point within the left-bottom most block (block 1) that has the least error (or the divergence at this point is closest to zero).

Handling Discontinuities and Critical Points

Critical points are important features of the flow datasets and their existence helps define the topology of the data. Invariably, most of the interesting flow datasets generally contain critical points. Although less prevalent, discontinuities can also be observed in the datasets. Examples of this can be an object inside the flow (flow around a cylinder datasets)

or presence of land in the ocean (Parallel Ocean Program datasets). Using our method, isolated critical points are automatically handled as will be shown in the Results section. For clusters of discontinuities, the parallel algorithm needs to be aware of the fact that flux computation can not propagate through those regions, instead it needs to go around the discontinuities as shown in Figure 6.4(b). Conceptually, this is similar to a flood fill algorithm starting from region A. Compared to the previously mentioned non-divergence regions, in this scenario, we only need to ignore the processing of the discontinuous regions. If there is a region that is surrounded by discontinuities, for example, region B in Figure 6.4(b), then this region is not synchronized with other regions but processed independently since no streamlines can leave this block.

6.2.4 Quality Analysis and Quantification

The method described in the previous sections is calculating a global field based on the local properties of the dataset. Since the final outputs from the proposed method are streamlines which are essentially isocontours of the stitched flux field, it is essential to provide the uncertainty factors in the analysis. The first important source of uncertainty for our case is the local numerical integration for computing the flux values for the blocks. Secondly, another source of uncertainty is the interpolation scheme for generating isocontours to show the streamlines. Higher order integration and interpolation techniques can be applied to reduce these two types of uncertainty without introducing much performance overhead. Since the particle-tracing based streamlines are also erroneous due to multiple factors [79], while checking the quality of our method, we do not directly compare with the streamlines; instead here we show the point-wise error of the generated scalar field and evaluate the quality. To achieve this, we construct a vector field from the scalar flux field

F by taking the local derivative at every grid point and check if it is everywhere normal to the origin vector field $\vec{V}$. Formally, our error measure is given as:

$$Err(F,\vec{V}) = (\nabla F) \cdot \vec{V} \tag{6.6}$$

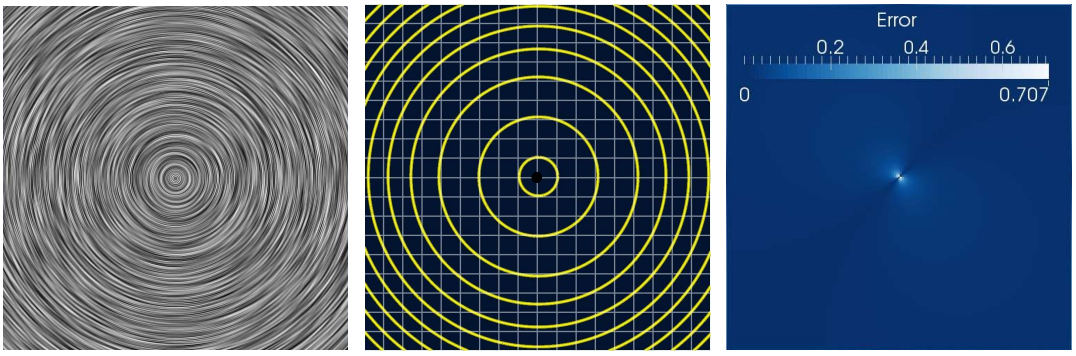
After generating this error field, we can visualize it to understand the error in our method. For a perfect reconstruction, the error should be 0 and the values closer to 1 will report a higher amount of error.

6.3 Results

Our experiments and scaling study were conducted on the Blue Gene/Q Vesta system at the Argonne Leadership Computing Facility. Vesta consists of 2,048 nodes where each node is equipped with 16 PowerPc A2 1600MHz cores and 16 GB of RAM. The total memory of Vesta is 32TB and it has a General Parallel File System. Next, we present the visualization results and the correctness of the generated streamlines to show the efficacy of our proposed method at varying block sizes.

6.3.1 Circle and Saddle Datasets

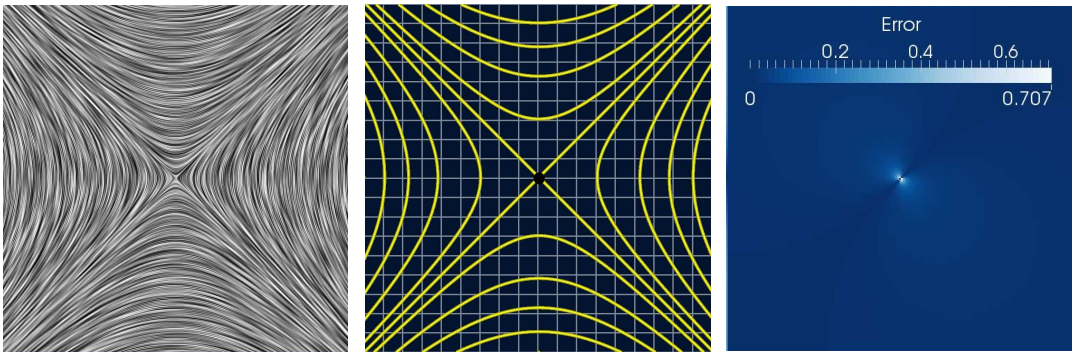
First, we apply our algorithm on two simple example datasets to demonstrate its effectiveness. These two datasets are generated using the code distributed by Turk and Banks [133]. Both the datasets were set to a resolution of 6400×6400. The first dataset Circle describes a circular flow. LIC visualization of the Circle dataset is presented in Figure 6.5(a). The other dataset, Saddle, describes a two-dimensional saddle structure and it is presented in Figure 6.5(d). Both the datasets contain critical points at the center. Initially, the two datasets are divided into multiple small data blocks and a flux field is generated using local flux based computation. A very similar result can be obtained even if the block



(a) LIC visualization of Circle dataset.

(b) Stitched streamlines from the Circle dataset.

(c) Error visualization of the streamlines from Circle dataset.



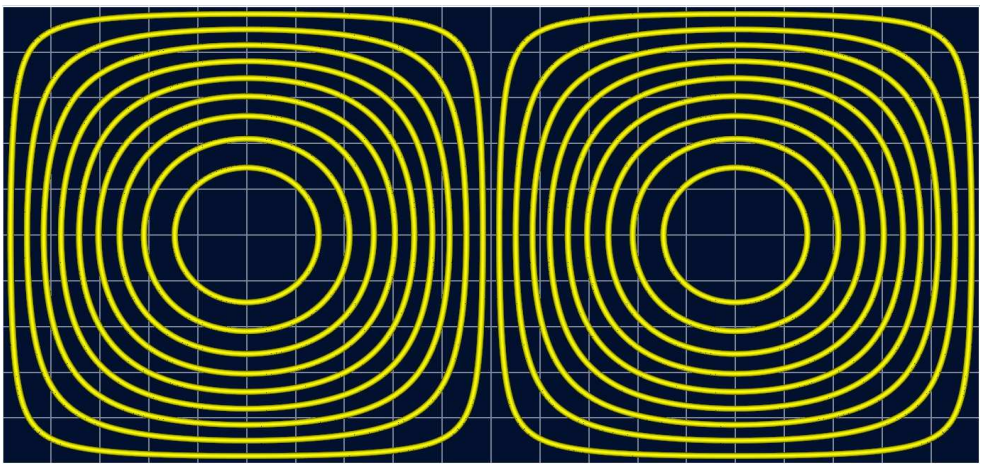
(d) LIC visualization of Saddle dataset.

(e) Stitched streamlines from the Saddle dataset.

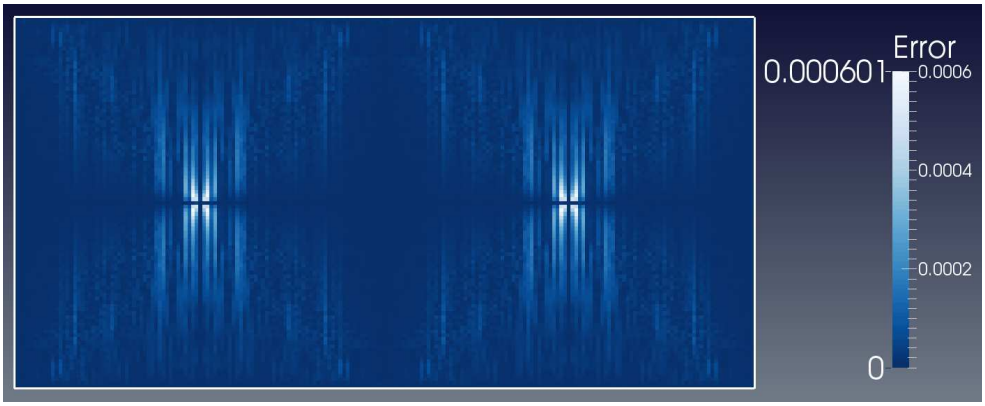
(f) Error visualization of the streamlines from Saddle dataset.

Figure 6.5: Stitched streamlines from Circle and Saddle datasets and their respective error visualizations.

size is varied. The local data blocks are then stitched across the blocks through the propagation of flux values and the results are presented in Figure 6.5. Figure 6.5(b) represents the stitching results from the Circle dataset where yellow curves are the isocontours in the stitched field that correspond to the streamlines of the field. The data subdivision is shown as white grids in the background of the figures. The quality of the proposed method on this dataset is shown in Figure 6.5(c) where the error is quantified according to the measure described in Section 6.2.4. Similarly, for the Saddle dataset, the stitched streamlines are depicted in Figure 6.5(e).The corresponding error visualization is shown in Figure 6.5(f).



(a) Stitched streamlines from the Double gyre dataset.



(b) Error of Double gyre dataset.

Figure 6.6: Stitched streamlines from Double Gyre dataset and error visualization.

In both the error visualizations, it is seen that most of the dataset has very low error and the highest error occurs near the center critical point of the datasets. From these results, it can be concluded that our proposed methods work quite well for these datasets.

6.3.2 Double Gyre

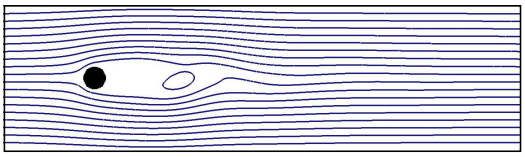
Next we show our result on the analytic dataset Double Gyre which contains two symmetric vortices. This dataset was generated using the following equations:

$$\begin{aligned} u(x,y) &= -\pi \sin(\pi x)\cos(\pi y) \\ v(x,y) &= \pi \cos(\pi x)\sin(\pi y) \end{aligned} \tag{6.7}$$

over the region $[0,2] \times [0,1]$. We test on the dataset that has a resolution of 2001×1001. To generate the results, we first divide the whole dataset into contiguous 101×101 blocks and compute the local flux fields for each block independently. Then starting from the left-bottom corner-most block, the flux values are propagated similar to what is shown in Figure 6.3(d). After the stitching process, the isocontours are computed. In Figure 6.6(a), we show the isocontours computed from the stitched flux field of this dataset and the data partitioning is shown in the background of the figure. From this figure, it can be observed that the maximum error is very small which reflects a good reconstruction of the vector field.

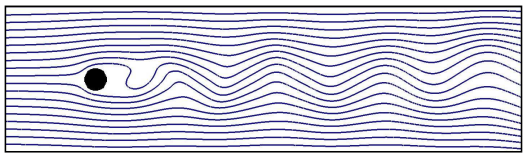
6.3.3 Flow Around a Cylinder

Next, we show the results on the dataset that describes a two-dimensional flow around a cylinder for multiple time steps. This dataset represents the Karman vortex streets in the fluid flow. From this dataset, we have chosen three time-steps at equal intervals: time step 30, 60 and 90 and applied our method. Each time step has a resolution of 768×231. For computation of the local streamlines within the blocks, the block size was chosen to be 11×11. Computation of divergence field on this dataset yields very small values that suggests that this dataset can be used as incompressible flow without much error. This



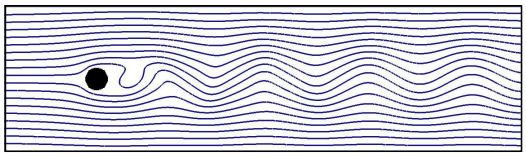
(a) Stitched streamlines from the time step 30 of flow around the cylinder.

(b) Vector field reconstruction error in the time step 30 of the flow around the cylinder.



(c) Stitched streamlines from the time step 60 of flow around the cylinder.

(d) Vector field reconstruction error in the time step 60 of the flow around the cylinder.

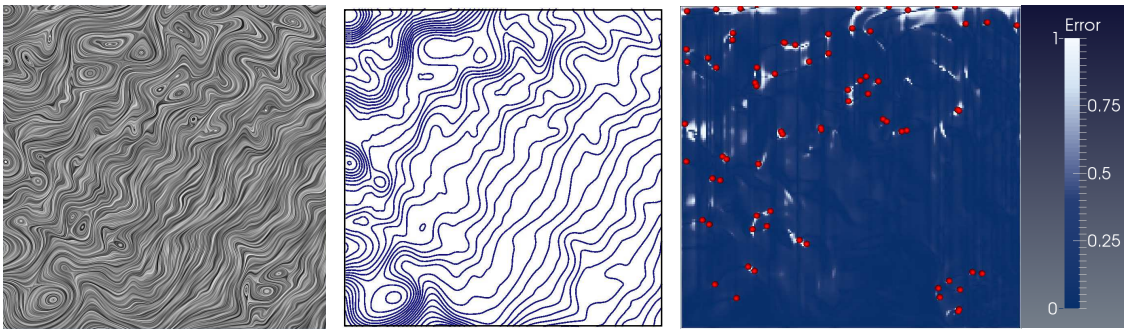


(e) Stitched streamlines from the time step 90 of flow around the cylinder.

(f) Vector field reconstruction error in the time step 90 of the flow around the cylinder.

Figure 6.7: Results for different time steps for flow around a cylinder dataset.

dataset contains discontinuities where the cylinder is placed inside the flow as shown by the black color in Figure 6.8. Figure 6.7(a) represents the reconstructed isocontours after the stitching across the blocks. The effectiveness of the reconstruction is shown in Figure 6.7(b). As can be seen in this figure, the flux field is in close agreement with the original vector field. The higher error regions are near critical points and regions of discontinuity. Further, Figure 6.7(c) and Figure 6.7(e) represent the isocontours from the time steps 60 and 90 with corresponding error visualizations provided in Figure 6.7(d) and Figure 6.7(f). In these figures, we again see that our local computation and stitching process was able to generate streamlines with very low amount of error.



(a) LIC visualization of the POP dataset.

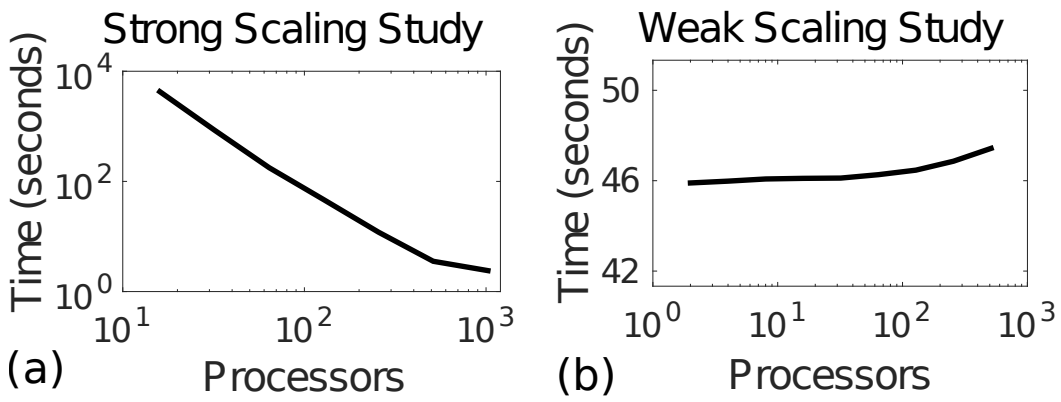
(b) Stitched streamlines from the POP dataset.

(c) Error visualization with the critical points shown.

Figure 6.8: Results for the POP dataset.

6.3.4 Parallel Ocean Program

Here we present the output of Los Alamos National Laboratory's Parallel Ocean Program (POP) model as our next dataset and show the results. The full dataset is three dimensional with a resolution of $3600 \times 2400 \times 42$ but scientists generally use this dataset slice by slice to explore the characteristics of the different regions of the sea. The velocity in the z direction is often ignored for general analysis and we have followed the same approach to produce our results. From the dataset, we have extracted a region near the land of the South American continent with a resolution of 200×200 and applied our method on it. A LIC visualization of this region is provided in Figure 6.8(a). From this figure, we can see that the flow in the selected region is quite complex and contains many critical points with high curvature regions. Now we divide the data into 11×11 chunks and apply our algorithm to generate the flux field with stitching. The isocontours of this stitched flux field is presented in Figure 6.8(b). By visual analysis, we can readily observe that the streamlines conform with the flow directions shown in the LIC image. To estimate the quality, we compute the



(a) (b)

Figure 6.9: Scaling study. (a) Strong scaling study performed on upsampled circle dataset. (b) Weak scaling study performed on various upsampled circle datasets.

gradient of the flux field and find the dot product with the original vectors and present in Figure 6.8(c). Also, the critical points are highlighted as the red points in this image to show the correlation of error and critical points of the field. As we can observe, although the quality overall is quite good, near the critical points and high curvature regions, we get higher error. We note that due to the non-negligible divergence values in some regions of this dataset, the flux differences across the blocks had some mismatch which was compensated by adding an extra flow amount in the block. Despite this flux mismatch, as can be seen from the resulting figures, the reconstruction has high accuracy in other regions and generated streamlines are smooth and represent the flow features very well.

6.4 Discussion

6.4.1 Scaling and Performance Study

To understand the scalability of our proposed system, experiments were conducted on multi-core machines and these results are presented in Figure 6.9. For the strong scaling

experiment, an upsampled version of the Circle dataset was used whose dimensions were 6400×6400 for the vector field. Number of processors were increased up to 1024 and total running time (total running time = local flux computation + communication + isocontour extraction) was recorded as shown in 6.9a. This log-log scale plot shows that our method scales well with the increase in number of processors. Figure 6.9b represents weak scaling study where again the Circle dataset was used. For generating the plots, data resolution and processor number were doubled starting from data size 400×400 with 2 processors up to data size 6400×6400 for 512 processors. In this weak scaling study, again a good scaling can be seen.

Next, we present the comparison of our flux based streamline generation method with a numerical integration based approach. For this comparison, we have used the popular OSUFlow library that implements Peterka et al.'s proposed method [102] of parallelizing streamline computation by distributing the data blocks across processors. This method is chosen for comparison instead of parallel-over-seeds since the execution model of large ocean simulations is also parallel-over-blocks. For integration based streamline generation, a fixed step size of one voxel length was used along with RK4 integration method. In our proposed method, the VTK [120] library was used to generate the final isocontours. We have used 128 processors and varied the generated streamlines from 10^3 to 10^5 in an upsampled version of flow around a cylinder dataset (with effective data size 4800×2000) and recorded the total running times in Figure 6.10. As can be seen from this figure, our method takes much less time than OSUFlow when more streamlines are traced. Since the execution speed of our flux-based method only depends on the data size rather than the complexity of vector data, we expect to see similar speed-up trends and scalability reports for other datasets of similar sizes. As can be seen from the scalability and performance

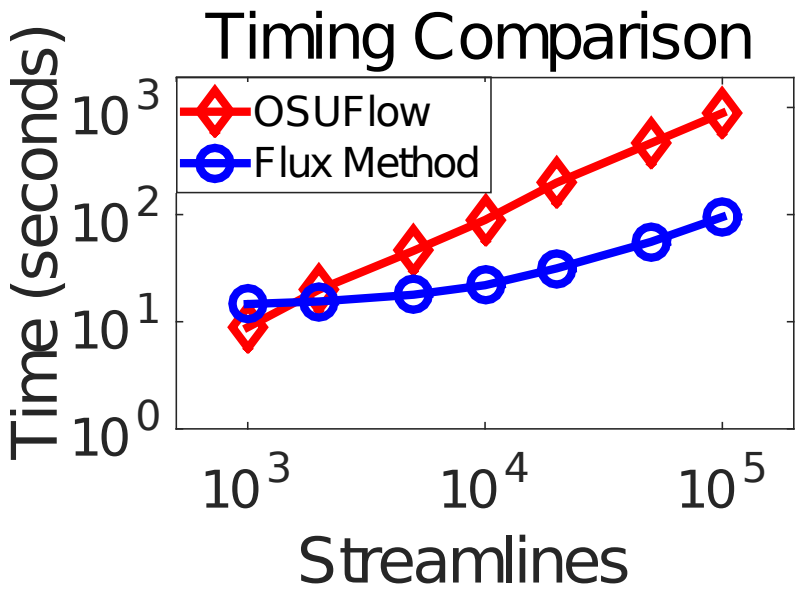


Figure 6.10: A comparison of performance between our flux based method and OSUFlow generated streamlines.

results, our method is well suited for the use cases where a large number of streamlines are to be generated.

6.4.2 Consistency

While generating long streamlines, generally numerical integration based approaches become less reliable due to the accumulation of local integration errors at each step. Reliability of numerical integrations vary depending on the integration method, step size and interpolation method used as discussed by several researchers [11, 19, 79, 94, 151]. Stream function based approaches generally provide a more reliable solution compared to the numerical integration based techniques. Apart from having a very low error in the reconstructed vector fields, one specific example can be shown to depict the consistency of our proposed method. As described by Perry *et al.* [101], in two-dimensional incompressible

flow, the critical points can only be of two types: center and saddle points. Now, if we start to trace a particle using Runge-Kutta 4th order (RK4) method in the flow around a cylinder data near a critical point (as shown in Figure 6.11a), the ideal behavior should be to get a close loop streamline. Instead, as depicted in Figure 6.11b, we see that the streamline is spiraling and tending outwards from the critical point. This behavior should be observed near a spiraling source but as we mentioned before, this dataset does not contain a source/sink. It is observed that, compared to Figure 6.11b, our method can produce more consistent results as depicted in Figure 6.11c.

6.4.3 Limitations and Future Work

As shown in the Results section, the proposed approach performs well for two-dimensional datasets that can be approximated as incompressible flows. In this section, we discuss the possible extensions of our method for three-dimensional fluids and compressible fluid flows. As mentioned before, the incompressibility assumption is quite popular in the CFD community and computation of divergence is an indicator to the incompressibility of the generated dataset. When the divergence is relatively low or close to zero for a dataset, our proposed method performs very well. If divergence is higher everywhere in the dataset, e.g. for a compressible fluid flow simulation, then the errors can become non-negligible. To formally propose a solution for compressible flows, the density can be considered in the mass flow computation. Following this approach, the datasets will require the density field along with the velocity fields for this method to work and this is one of our future extensions to this current algorithm. We note that, if the flow does not have a z-component or if z-component can be ignored like the POP dataset, then even three-dimensional datasets can also be handled by applying our algorithm on each slice. For extending to general

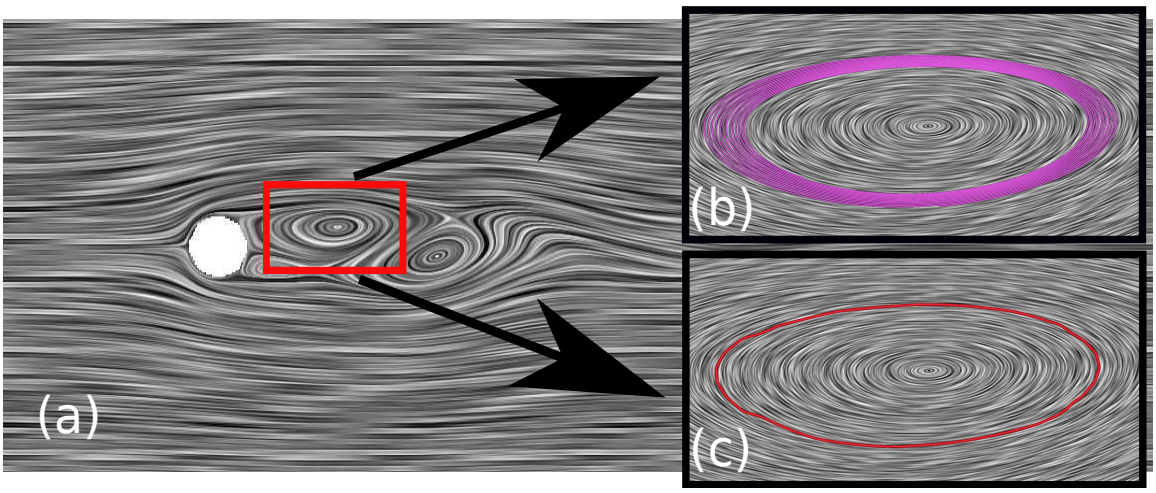


Figure 6.11: A comparison of the RK4 method and our flux based method. (a) LIC visualization of the time step 30 of flow around a cylinder data where a region near a critical point is selected. (b) Streamline generated using the traditional RK4 integration incorrectly suggests the existence of a spiraling source. (c) Our flux-based approach correctly highlights the existence of a center with a closed loop streamline.

three-dimensional datasets, our algorithm needs to be modified such that it generates a three-dimensional flux field. The motivations for this can be taken from three-dimensional stream function computations which are computationally involved [50, 60, 88] but are possible to compute with some constraints and assumptions. Apart from extending our method to three-dimensional compressible flows, we also plan to extend this method for uncertain datasets as part of our future work.

6.5 Conclusions

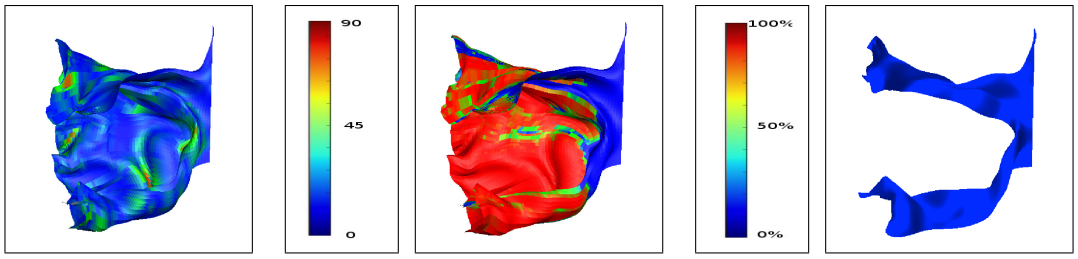
In this work, we proposed a novel technique for generating streamlines in parallel with high scalability for large two-dimensional flow fields. Instead of applying the traditional numerical integration schemes on the vector fields, we propose to compute a flux-based scalar field for implicit streamlines in parallel. Initially, data is subdivided into smaller

blocks and local flux field is generated whose isocontours represent local streamlines. Since each block computes flux values according to a local flux origin, alignment of flux origins is needed to generate coherent streamlines across blocks. To achieve this, we use the additive property of flux and each block sends its local offset to its neighbors. Since this communication stage involves sending one float value across the neighbors of each block, it is efficiently performed and a stitched flux field is generated. Finally, isocontours of this stitched field are extracted in parallel to produce the final streamlines. The results obtained from a variety of datasets, both analytical and simulated, show the effectiveness of the stitching method. Our experiments also reveal good strong and weak scaling properties and much-improved execution time.

Chapter 7: Error Quantification Metrics for Evaluation of Stream Surfaces

Similar to the streamlines discussed in the previous chapter, stream surfaces are also extensively used for flow field exploration. Stream surfaces are extensions of streamlines, where the normal direction of any point on the stream surface is perpendicular to the underlying flow field. As they provide better 3D depth cues, these surfaces can be more powerful in revealing flow features, although they are also more prone to issues like occlusion. Compared to the computation of streamlines, generation of stream surfaces is less straightforward and is more susceptible to discretization errors. To date, there exist several algorithms designed to address various issues to improve the quality of stream surfaces. While the relative difficulty in construction of stream surfaces has led researchers to look for algorithms [115, 90] that are less complex, generating a surface that faithfully represents the underlying flow, has remained the basic requirement of any stream surface generation algorithm.

Although stream surface is a popular visualization tool among scientists, a thorough discussion of the stream surface errors is mostly lacking and the need of a stream surface error measurement technique is ubiquitous. Figure 7.1(a)-(e) can be used as one of the motivations for coming up with the error metrics for stream surfaces. Ignoring the color mapping on the surfaces, if the user only observes the stream surface as shown in Figure



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

Figure 7.1: (a) Sampled local error visualization for a stream surface. (b) Color map shows the average angle deviation (in degrees) for the surface patches. (c) Sampled global error visualization for the same stream surface. This metric reveals the erroneous regions not shown by the local error metric. (d) Color map shows the percentage of the sample points which miss the seeding curve by more than a threshold value when traced back. (e) Streamline simulation of the stream surface with the same seeding curve as (a) and (c).

7.1(a) and Figure 7.1(c), the user may not know that this stream surface has large erroneous regions and the topology of the surface is not correct. In this example, Figure 7.1(c) shows the erroneous regions and Figure 7.1(e) shows the correct topology for the surface. Using the metrics proposed in this chapter, we quantify the geometric errors present in the stream surfaces.

Apart from revealing the level of uncertainty of the stream surface visualizations, these error metrics also help in the exploration of the multi-dimensional parameter space associated with the stream surfaces. For generating stream surfaces, there exist many algorithms at this moment and choosing one of them is just one of the dimensions. Then, choosing where to place the initial seeding curve is another very important aspect as the experimental results show that, depending on where the stream surface is seeded, different algorithms behave differently. The next decision problem is the seeding curve sampling density which affects the final output from different algorithms in different ways. After these three steps, the user needs to decide the parameters used by the stream surface algorithms for finally

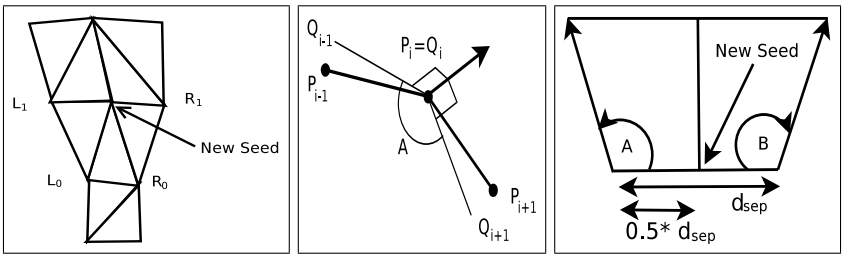
generating the surface. In this exploration process, the need for stream surface quality metrics is self-evident although this kind of work is mostly lacking up to this point. In this chapter, we put forward four different methods to evaluate the stream surfaces quantitatively. With the metrics, we explore the different dimensions attached to the stream surface generation process with three popular stream surface generation algorithms, namely Hultquist's algorithm [61], Garth et al.'s algorithm [45] and McLoughlin et al.'s [90] algorithm. To the best of our knowledge, there exist very few research works that discuss stream surface verification with error quantification and visualization.

7.1 Background

Stream surfaces are surfaces where all points are always tangential to the flow field. This is formally described [61] as a two-dimensional parametric surface that is embedded in the three-dimensional flow. This parameterization can be done along parameters s and t. Using the parameter $s \in [0,1]$, we can parameterize the streamlines according to their seed locations. The second parameterization is done for time $t \in [t_0, t_{max}]$ along every streamline. A constant s curve is a streamline and a constant t curve is a timeline. A stream surface basically consists of an infinite number of streamlines which are seeded from the seeding curves. When calculating stream surfaces, we assume that the flow field is time-invariant. If the flow field is time-varying, the surface is called a path surface, which is a natural extension of a stream surface.

7.1.1 Stream Surface Algorithms

There exist several stream surface generation algorithms, and we select the following three algorithms as the representative ones to study: Hultquist's method [61], Garth et al.'s method [45] and McLoughlin et al.'s method [90]. Hultquist's algorithm is one of



(a) (b) (c)

Figure 7.2: New seed insertion : (a) Hultquist's algorithm : When divergence is detected, new particles are added to generate additional triangles. (b) Garth's algorithm : High curvature regions are detected by the neighboring ribbon angles and new seed is inserted. (c) Quad algorithm : When $A > 90^\circ$, $B > 90^\circ$, $d_{sep} >$ half the data sampling points, then a new seed is inserted.

the earliest stream surface algorithms and most other stream surface algorithms are based on it. Garth's algorithm is generally regarded as the best stream surface algorithm for handling high curvature flows. McLoughlin's method generates quads instead of triangles and is relatively easy to implement. Below we briefly review these three stream surface algorithms.

Hultquist's Method : This algorithm generates a triangular mesh. The seeding curve is sampled and streamlines are generated from these seed points. Two neighboring streamlines are then tiled according to a locally optimal greedy tiling method. This method checks the leading edges of the two possible triangles and picks the one with the minimum length for triangulation. Divergence is detected when the width of the current quadrilateral becomes twice the height of it. A new seed is then inserted at the midway point between the two most recent points of the bordering curves. Figure 7.2(a) shows one splitting scenario where the quadrilateral $L_0R_0R_1L_1$ is split into three triangles for a new seed insertion. In merging flow, the quadrilaterals of the two neighboring ribbons are checked if they are

roughly coplanar and their combined width is smaller than their height. Then the middle particle is deemed unnecessary and removed. Ripping is detected when two neighboring streamlines are traveling in almost opposite direction.

Garth's Method : Garth's algorithm inherits the basic idea of the Hultquist's algorithm, but with modifications that are more suitable for turbulent flows. This algorithm performs arc length parameterization of streamlines. Also, a new rule is added for new seed insertion. The angle between the two neighboring ribbons, A, is checked where angle $A = \angle(\overrightarrow{Q_{i-1}Q_i}, \overrightarrow{Q_iQ_{i+1}})$ where points $Q_j, j \in \{1,2,..,n\}$ are the projections of the front node points $P_j, j \in \{1,2,..,n\}$ into the plane perpendicular to the flow at that node point P_i. If this angle exceeds the maximum threshold and if the maximum front resolution is not yet reached, then a new node is inserted. A node is deleted if this angle falls below a minimum threshold. We keep the maximum and minimum allowed angular deviations set to 15 degrees and 5 degrees respectively. Figure 7.2(b) shows the angular criterion for new seed insertion which is similar to the figure shown in [45].

McLoughlin's Method or Quad Method : This method generates quads instead of triangles. At each iteration over the seed points, it integrates the points, and quads are generated which are then rendered. Divergence is detected when the base of the quadrilateral is greater than half of the data sample points and the internal angles of the quadrilateral incident on the base are greater than 90°, at which point a new seed is introduced. When the combined length of the two neighboring quadrilaterals' bases are smaller than the data sample points, and the quad internal angles incident on the base are smaller than 90°, then the middle point is removed. When high curvature is detected, the quad streamline edges are shortened to maintain the shape. Figure 7.2(c) shows a typical split scenario for this case.

7.1.2 Error in Stream Surfaces

When streamlines are constructed, the main source of uncertainty comes from the error incurred in the numerical integration step. Depending on the integration scheme in use, we can have a theoretical bound on the error. Since a stream surface is ideally thought of as a collection of streamlines, this integration error is present also in stream surfaces. Another important factor is the sampling decision. Generally, stream surface algorithms start computing streamlines from points on the seeding curve. This seeding curve sampling density is normally not enough, so the algorithms insert new seeds as the surface is being constructed. As described in the three algorithms above, the insertion of new seeds is performed based on heuristics. Since there is no guarantee that these new seeds are actually originating from the seeding curve, there are chances of more error being introduced in the surface. We call this phenomenon the uncertainty due to insufficient sampling of the seeding curve. Here we assume that the use of high order Runge-Kutta methods makes the integration error quite small. We are primarily concerned with capturing the error due to insufficient sampling of the seeding curve and its effect over the generated stream surface. The metrics to quantify error are lacking, although the need for this is self-evident.

7.2 Error Quantification Methods

In this section we present four methods to verify stream surfaces, each originating from the definition of an ideal stream surface and covering a particular aspect of error in stream surfaces.

7.2.1 Sampled Local Error Method

Most of the existing stream surface generation techniques produce a triangle/quad mesh as the output, which is then used for rendering. From the stream surface definition, all the points in a given surface patch should be tangential to the underlying flow, or in other words, the surface normal of that given patch should be perpendicular to the flow direction. This gives a way of calculating the error in a stream surface patch. The idea is to sample the given stream surface patch, find N sample points randomly distributed over the region, and then for each point, evaluate the dot product of the unit surface normal and the unit flow vector. Since for a perfectly error free region, the values should be all zeros, we can look at the distribution of the dot product values to determine the quality of the patch. We call this error estimation technique "local" as this error metric only measures the quality of the given surface patch in that region of flow. If the stream surface has deviated from the ideal location starting from the seeding curve, this method may not detect that error.

For algorithms which generate triangular meshes, finding the surface normal is relatively straightforward. Since any point $\mathbf{r}$ that is inside of the triangle can be expressed in barycentric coordinates, N sample points can be found by generating random values between 0 and 1 for α and β and then checking whether $0 \leq \gamma \leq 1$ where $\gamma = 1 - \alpha - \beta$. If $\mathbf{V}_r$ is the normalized velocity vector at position $\mathbf{r}$, the dot product of $\mathbf{V}_r$ and the surface normal is calculated for N different sample positions and a distribution of values is generated. For a good quality triangle, the mean of these values should be close to 0 with a small standard deviation.

On the other hand, the quads generated by stream surface algorithms, are not guaranteed to be coplanar, e.g. in case of a complex flow region, McLoughlin et al.'s method produces quads whose four vertices are not necessarily in the same plane. If four points of a quad are

not coplanar, then an equation of the plane, which describes the quad, cannot be found and hence generating samples on the quad would be less straightforward. In our method, each sample point within the quad was found by using two random numbers which were used for bilinear interpolation between four vertices. To get the normal at the sample point, we first calculate the vertex normal per quad vertex and then apply bilinear interpolation to calculate the normal at this sample point from the vertex normals. The same two random numbers were also used here for bilinear interpolation. Then the dot product can be computed. A local error visualization is shown in the Figure 7.1(a)-(b). Here the error is measured as the average angle deviation of the surface patches and is color mapped. For any algorithm that produces triangular or quad mesh, this method can be used to visualize the local error.

7.2.2 Sampled Global Error Method

The previous method suffers from the drawback that if the stream surface has drifted off from the ideal stream surface, the local metric will not detect it as it only checks whether the surface patch is correct with respect to the local flow. A global error check enforces that the surface patch is locally correct and it originates from the given seeding curve. Our global error calculation method finds sample points on a patch of the surface and backtraces them to find out how many of these samples go back to the seeding curve. If the sample point of the generated surface results in a streamline connecting it to the seeding curve, then it can be assumed that the sample point is of good quality. If we assume that the integration error in the streamline computation is small, then this method yields a good verification result.

The process of finding the sample points on the mesh remains the same as the method described above. While sampling the quads/triangles, the number of forward integrations done up to that point, T, is known. For integration step size S, initially the sample points

are backtraced $T-1$ steps and then it is backtraced M times with step size S/M and the minimum distance obtained at any step, is stored. M is set to 20 when S takes the value 1.0 voxel size. These minimum distance values from sample points are gathered to get a distribution and the mean and standard deviation of this distribution of values should be close to 0 for a good quality surface. For quads, this backtracing can be sped up by using the (s,t) parameterization associated with each sample point where the parameterization can be used to predict the integration step size within the quad. Figure 7.1(c)-(d) shows a global error study for the stream surface shown in Figure 7.1(a)-(b). Here, 60 random sample points for each surface patch are chosen and error for each surface patch is measured as the percentage of the sample points that miss the seeding curve by more than a small threshold value when tracing back. The threshold value is set to 0.5 voxel distance in our experiment. It is observed that, in the particular case shown in Figure 7.1(a)-(d), the local metric was not able to capture the erroneous regions of the surface but the global metric successfully highlighted the error.

7.2.3 Densely Seeded Streamlines Method

Conceptually, a stream surface can be thought of as a collection of streamlines which are seeded infinitely densely from the original seeding curve. But practically it is impossible to sample the seeding curve infinitely densely. We simulate this by putting the seeds very densely on the seeding curve. Without considering numerical integration error, this collection of streamlines should be close to the ideal stream surface. Figure 7.1(e) shows a streamline simulation of the stream surface whose initial seeding curve is placed at the same region as the ones shown in Figure 7.1(a) and 7.1(c). It is evident that it shows the

region that the stream surface should be passing through compared to the previously generated stream surface which had erroneous regions. This method is similar to the work of Garth et al. [44] with an important difference. In their work, the authors assumed the collection of the streamlines to be the ground truth but our experiments involving the datasets having complex flows, reveal that it is not always safe to assume that the collection of streamlines will be representing the true surface. If the number of integration steps is large and the underlying flow is very divergent, then finding out the suitable seeding curve density is non-trivial as the neighboring streamlines will start to diverge. Given this fact, we take this collection of streamlines as a subset of the true stream surface and compare with the surfaces generated by the algorithms.

Comparison of the two stream surfaces, one generated from the densely seeded streamlines and one from an existing stream surface algorithm, is also non-trivial. In our method, the Hausdorff distance [4] is used to measure the difference between the two surfaces. Given two non-empty sets P and Q, the Hausdorff distance between them $d_H(P,Q)$ is defined as

$$d_H(P,Q) = max(\hat{d}(P,Q),\hat{d}(Q,P)),\ where\ \ \hat{d}(P,Q) = \max_{x\in P}\min_{y\in Q}||x-y||. \qquad (7.1)$$

Specific to this case, $||x-y||$ is taken as the Euclidean distance and one-sided Hausdorff distance from the streamline set to the algorithm generated stream surface is used. The Hausdorff distance calculation was sped up using GPUs and the comparison of two sets containing 1M points can be done within 10 seconds. A small value of this metric will indicate a high-quality stream surface. But if the Hausdorff distance value is large, then it may be difficult to compare the results of two algorithms because the large distance may be caused by a small number of outliers in the sample. To avoid this, we look at the

distribution of the minimum distance values and make a conclusion about the performance of the algorithms.

7.2.4 All Vertex Backward Tracing Method

No matter how densely the seeding curve is sampled, in the case of a complex flow, there will be cases where the neighboring streamlines will start to diverge. This scenario is also encountered in forward stream surface generation algorithms which is handled by putting more seeds in between two neighboring streamlines. Since these seeds are not guaranteed to be coming from the seeding curve and, even with many streamlines, there are chances that some features might still be missed. We can avoid this by doing backward integration from all points in space. All point backward tracing was previously employed by Van Wijk [135] in his implicit stream surface computation. The natural discretization of the flow field is given in the form of grid points. A way of checking the quality of a stream surface would be to trace out streamlines in the backward direction from all grid points. The points which trace back to the seeding curve are the points which should be included in the stream surface. As we cannot sample the flow field infinitely, every grid point can be checked to find out the closest distance of that backtraced point to the seeding curve. As the number of forward integration steps is known for a given stream surface, each grid can be backtraced those many steps and any grid point that comes closer than a predetermined threshold value while backtracing, will be included in a set that stores the candidate grid points to be included in the stream surface. The threshold can be set to a small value like 0.5 or 1.0. This way, a new collection of points will be formed. The one-sided Hausdorff distance can be calculated between this point set and the generated stream surface point set. If the resulting value is low, then the generated stream surface will be considered to be

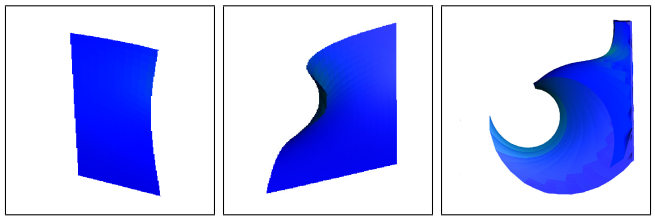
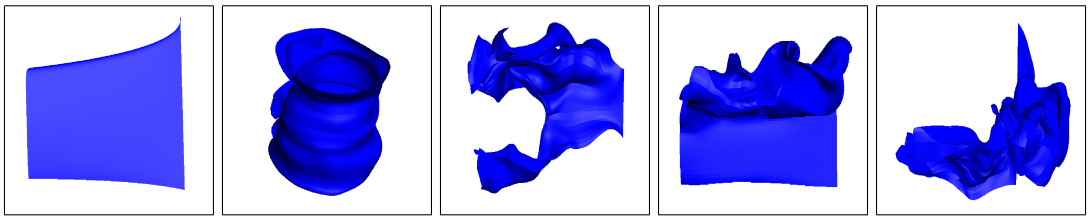


Figure 7.3: Comparison of three stream surfaces based on Hausdorff distance metric.

accurate. As the grid resolution is increased, this metric will perform better in estimating the quality.

The leftmost stream surface shown in Figure 7.3(a) has a simple straight underlying flow structure. The forward method of Section 4.3 gives the Hausdorff distance value 1.05 and backward method of Section 4.4 gives the Hausdorff distance value 1.2. The surface of the Figure 7.3(b) has a more complicated flow structure. In this case, the forward method gives 3.08 and the backward method gives 2.05. So it signifies that it has a higher amount of error than the previous one. In the stream surface of Figure 7.3(c), we have a more complicated stream surface from a tornado dataset. In this case, the forward metric gives 6.07 and the backward metric gives 6.8.

Although the forward and backward metrics produce similar results, it is important to see when they should be used. If the dataset is small, then the backward method may give us higher performance, as there are fewer streamlines to be computed as compared to the forward method. But if the dataset is large, it may be a better idea to use the forward method, which may calculate a smaller amount of streamlines and be more efficient.



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

Figure 7.4: Five stream surfaces showing different cases: a) StreamSurface1 showing straight areas of the flow, b) StreamSurface2 showing the vortex structure and high curvature areas of the flow, c) StreamSurface3 showing a general complex stream surface, d) StreamSurface4 has complex flow in the top part and straight flow in the lower region, e) StreamSurface5 is seeded in more turbulent region.

7.3 Results

The experiments were conducted on a Linux machine with the Intel core i7-2600 CPU and 16 GB of RAM. The two datasets used were the Plume and the Isabel. The Plume dataset is a simulation of the thermal downflow plumes on the surface of the sun and it has $126 \times 126 \times 512$ grid points. The dataset Isabel is a hurricane simulation with a resolution of $500 \times 500 \times 100$. To illustrate different seeding curve locations, five representative stream surfaces have been chosen, as presented in Figure 7.4(a)-(e). StreamSurface1 represents a very straight flow region of the field from the Plume dataset generated by integrating 100 steps, StreamSurface2 is the hurricane eye from the Isabel dataset, generated by integrating 150 steps, and is representative of the vortex-like structures which is quite common in flow fields, StreamSurface3 represents a general stream surface from the Plume dataset, generated by 100 integration steps, which has a complex underlying flow, StreamSurface4 has both complex and straight flow regions from the Plume dataset and it was generated by 100 integration steps, and StreamSurface5 is seeded in a complex region and has a very

complex structure from the Plume dataset and this was generated by 50 integration steps. We implemented the three stream surface algorithms for our experiments and adaptive Runge-Kutta Four-Five integration scheme was used for all three algorithms. The datasets were also normalized so that these algorithms could be compared on a common base.

7.3.1 Effect of the Choice of the Stream Surface Generation Algorithms and Placement of Initial Seeding Curve

In this section, the effect of choosing the stream surface algorithm for different initial seeding curve placements is presented. The three algorithms are tested against the five stream surfaces based on the four error metrics proposed in this chapter. The evaluation based on the four different metrics is presented separately in the following sections.

Results for the Sampled Local Error Method

In Table 7.1, we show the result of the application of the sampled local error metric as described in Section 4.1. The average angle deviation for each triangle/quad was measured by the average dot product value of each triangle/quad and then the final weighted average was calculated by taking the areas of the triangle/quad as the weights. Finally, the result is expressed in degrees . The area was used as the weights while averaging since different algorithms produce different numbers of triangles/quads. For each quad/triangle, 60 random samples have been used. Garth's algorithm still performing quite well in comparison. Finally, for StreamSurface5, Hultquist's algorithm falls behind with quad algorithm and Garth's algorithm yielding better average angle deviations.

Results for the Sampled Global Error Method

In the Table 7.1, we also represent the results of the application of the second metric, the sampled global error metric, as described in Section 4.2, for the five chosen surfaces.

Table 7.1: Results for the Sampled Local Error Method and the Sampled Global Error Method.

	StreamSurface1		StreamSurface2		StreamSurface3		StreamSurface4		StreamSurface5	
	Local Error	Global Error	Local Error	Global Error	Local Error	Global Error	Local Error	Global Error	Local Error	Global Error
Hultquist	0.15	0.03	3.02	0.072	8.51	2.17	6.18	3.54	14.51	4.81
Garth	0.15	0.03	2.74	0.055	6.79	1.67	5.42	2.95	12.94	3.51
Quad	0.15	0.02	2.91	0.062	8.17	1.82	7.54	4.1	13.22	3.71

Table 7.2: Results for the Densely Seeded Streamlines Method. H-dis and Avg. stand for Hausdorff Distance and Average respectively.

	StreamSurface1		StreamSurface2		StreamSurface3		StreamSurface4		StreamSurface5	
	H-dis	Avg.	H-dis	Avg.	H-dis	Avg.	H-dis	Avg.	H-dis	Avg.
Hult.	2.01	0.27	5.51	1.32	19.33	0.93	17.72	2.57	14.39	1.85
Garth	2.01	0.27	4.02	0.77	13.75	0.61	16.77	2.09	10.59	1.12
Quad	2.01	0.28	5.26	1.26	18.87	0.81	17.89	2.95	12.41	1.44

We collect the closest distances of all the sample points from the seeding curve after the sample points are backtraced. The final average value is also weighted by the area of each triangle/quad. For each quad/triangle, 60 random samples have been used. better compared to quad algorithm but Garth's algorithm is again relatively better. For the fifth surface, it is seen that quad algorithm and Garth's algorithm again out-perform Hultquist's algorithm. It is also noted that as a stream surface gets more complicated, this average minimum distance value also increases. These results are quite similar to the results given by the previous method and it is seen that Hultquist's algorithm gives worse values when compared to two other metrics in complex flow areas in general. But, as shown by StreamSurface4, it can be better than the quad algorithm in certain situations.

Table 7.3: Results for the All Vertex Backtracking Method. H-dis and Avg. stand for Hausdorff Distance and Average respectively.

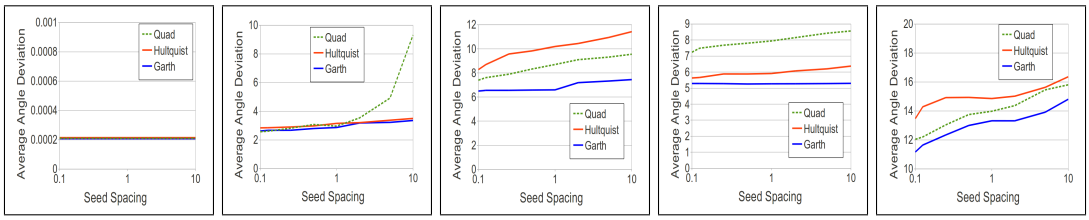
	StreamSurface1		StreamSurface2		StreamSurface3		StreamSurface4		StreamSurface5	
	H-dis	Avg.	H-dis	Avg.	H-dis	Avg.	H-dis	Avg.	H-dis	Avg.
Hult.	1.69	0.38	8.31	0.85	17.52	1.81	19.94	4.92	13.67	1.29
Garth	1.69	0.38	6.68	0.44	13.75	1.18	19.44	4.78	9.81	0.95
Quad	1.69	0.34	7.72	0.71	16.38	1.26	20.13	5.31	10.21	1.12

Results for the Densely Seeded Streamlines Method

In this section, we discuss the results of the three stream surface algorithms when compared against a point cloud which is created using the set of streamlines by sampling the seeding curve very densely. Now, with this point set, the outputs of the three algorithms are compared, and the minimum distances are collected for each point of the point set. The one-sided Hausdorff distance is calculated as the maximum value by which the points on the streamlines differ from the stream surface points. Also, the mean of these minimum value distribution are collected and the complete result is shown in Table 7.2. method, according to this metric. In case of StreamSurface4, Garth's algorithm does the best again but Hultquist's algorithm has done better than the quad algorithm. Finally, for StreamSurface5, it is found that Garth's algorithm and quad algorithm generate results which are better than Hultquist's algorithm.

Results for the All Vertex Backward Tracing Method

Here we discuss the results of the application of all vertex backtracing method. From all grid points of the flow field data, a streamline is traced back and at each step, it is checked to see how close it gets to the seeding curve. The cutoff threshold distance is kept as 0.75 and whenever a backward streamline gets any closer to the seeding curve than



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

Figure 7.5: Effect of initial seeding curve sampling density on the five stream surfaces for the three stream surface generation algorithms.

the cutoff distance, the grid point is marked. All these marked points form the point set that gives the rough outline of the ideal stream surface. From these points, we find the minimum distances to the point set generated by the existing stream surface algorithms. The one-sided Hausdorff distance and mean are listed in Table 7.3.

7.3.2 Effects of Varying Seeding Density of the Seeding Curve

In this section, initiative is taken to observe how the initial seeding curve sampling density influences different stream surface algorithms. Experiments have been conducted by varying the seeding density from one seed every 10 voxels to one seed every 0.1 voxel size for the previously mentioned five stream surfaces. In this case, as all the error metrics are showing the same trend, the sampled local error metric is chosen to show the results in Figure 7.5(a)-(e). It is observed that, if the stream surface is very flat, similar to StreamSurface1, seeding density does not affect the quality that much for all the three algorithms. If the stream surface gets complicated gradually, as StreamSurface3 and StreamSurface4, then the algorithms get the chance to dynamically adjust the seeding density by adding new seeds in the first few steps. In this case, although the amount of error increases as the seeding density decreases, this increase is quite moderate as shown in Figure 7.5(c)- (d).

StreamSurface5 was seeded in a very complex flow region, and although the algorithms tried to modify the seeding density in the first few steps, still the error incurred in those steps were quite high and the effect of sparse seeding is evident in Figure 7.5(e). Figure 7.5(b) is the case where quad algorithm performs really bad as the seeding density is decreased as compared to the other two algorithms. In this case, quad algorithm is not able to generate new seeds which other two algorithms could because quad algorithm's inner angle criterion is not satisfied while the surface generation and large quads have been produced.

7.3.3 Effect of Parameter Choice of Algorithms

In this section, we investigate the effect of parameters for the three algorithms using the sampled local error metric. In Hultquist's algorithm, the new seed insertion is done based on the ratio of the width and height of a quadrilateral while generating the mesh. Figure 7.6(a) shows the variation in quality for different choice of this width-to-height ratio threshold. For StreamSurface1 and StreamSurface2, there is little variation in quality

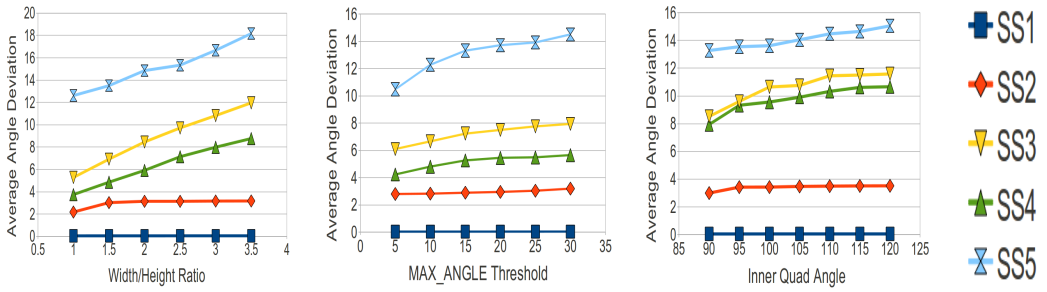
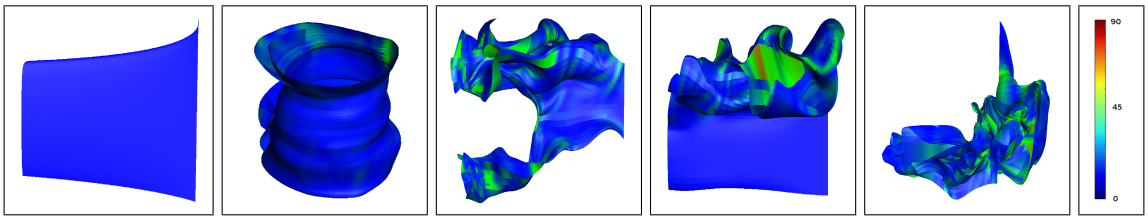


Figure 7.6: Effect of the user defined parameters: (a) Different values of widht-to-height ratio, used as the criterion for new seed insertion in Hultquist's algorithm. (b) Different Maximum Angle Threshold for Garth's algorithm. (c) Different inner angle threshold, used for detecting divergence in quad algorithm. (d) Legend

and number of triangles generated remain almost the same except at a very low ratio, unnecessary triangles are generated which do not aid in improving the quality that much. For StreamSurface3 and StreamSurface4, as the ratio threshold is decreased from 3.5 to close to 1, quality increases but also the triangle count goes up around 5 times from around 30K triangles to 150K triangles in both the cases. For StreamSurface5, the increase in triangle count is around 50 times from 33K to 1700K.

In Garth's algorithm, there are two user-defined parameters, namely the maximum angle threshold and minimum angle threshold which govern seed insertion and deletion. Here, we chose to vary the maximum angle threshold from 5 degrees to 30 degrees keeping the minimum angle threshold fixed at 2 degrees. The effect of this parameter is shown in Figure 7.6(b). It is observed that the quality of StreamSurface1 and StreamSurface2 remain almost unaffected by the change of this parameter although the triangle count varies from 25K to 1800K for StreamSurface2 when the angle threshold varies from 30 degrees to 5 degrees. StreamSurface3 and StreamSurface4 show that as the parameter is decreased, the quality increases with an increase of triangle count from 66K to 1150K and 61K to 1200K for the two surfaces respectively. StreamSurface5 shows the most variability with this parameter. In this case, the triangle count variation is 360K to 16300K.

For the quad algorithm, the split criterion is studied by changing the threshold angle from 120 degrees to close to 90 degrees and the result is shown in Figure 7.6(c). For StreamSurface1 and StreamSurface2, the variation in quality is relatively small and also the quad count remains almost the same for the two cases. For StreamSurface3 and StreamSurface4, the quad count is almost doubled from around 6K to 13K for both the cases as the



(a) (b) (c) (d) (e) (f)

Figure 7.7: Visualization of sampled local error for StreamSurface1 to StreamSurface5 respectively in (a)-(e). Surfaces are generated using quad algorithm. (f) Color map showing the average angle deviation (in degrees) mapped to color.

threshold is changed. For StreamSurface5, the change in quad count relatively less compared to the previous two surfaces. The quad count changes from 6.5K to 9.5K and also, the quality does not show a very big variation for different threshold values.

7.3.4 Error Visualization in the Stream Surfaces

The sampled local error and sampled global error metrics can be used for effective error visualization in the stream surfaces. In this section, the quad algorithm has been selected for the illustration of the error visualization for StreamSurface1 to StreamSurface5 and the sampled local error metric is chosen as the error quantifier, although, any other algorithm and sampled global error can also be chosen for error visualization purposes. As shown in Figure 7.7(a)-(e), the error is color mapped to represent the quality of the surface patch. The color map is shown in Figure 7.7(f) which shows the mapping of average angle deviation of the surface patch in the local region with respect to the local flow direction.

7.3.5 Computation Time and Mesh Element Count

Each stream surface algorithm takes different amount of time to generate stream surfaces. Also, they generate different amounts of triangles/quads. The statistics taken from

these algorithms for the given three stream surfaces is listed in Table 7.4. Besides the error computation, the computation time and generated triangle/quad count is also an important factor. If two algorithms are performing equally well in terms of error, then the faster one is probably preferred. From the data of Table 7.4, it can be seen that the quad algorithm is the fastest of the three algorithms. The quad algorithm also produces fewer quads, which is also an expected result.

Table 7.4: Performance (columns named Time) and triangles/quads (columns named T/Q) generated by the three algorithms.

	StreamSurface1		StreamSurface2		StreamSurface3		StreamSurface4		StreamSurface5	
	Time (ms)	T/Q (K)	Time (ms)	T/Q (K)	Time (ms)	T/Q (K)	Time (ms)	T/Q (K)	Time (ms)	T/Q (K)
Hultquist	749	16	1412	19	2768	59	2027	48	3921	97
Garth	830	16	2849	85	5370	119	4441	101	34210	739
Quad	285	7	650	10	970	14	465	13	255	9

7.3.6 Results Summary

It is observed that the results of the different stream surface verification metrics, agree on choosing the best algorithm for a given case. From the results, it is also apparent that for less complex flow regions, Hultquist's algorithm is almost as good as the other two algorithms although it tends to get worse when it encounters a more complex flow region. In high curvature regions, Garth's algorithm and the quad algorithm generally perform better than Hultquist's algorithm. These two algorithms use angles as their ribbon splitting criteria, which makes them better suited to handle complex flows. But there is an instance where Hultquist's algorithm outperforms the quad algorithm. Quad method, although takes the least time and generates good quality surfaces, it should be chosen carefully as its behavior

in certain conditions is less reliable as shown by the previous results. Garth's algorithm consistently performs well but takes more computation time and produces larger amount of mesh elements compared to the other two algorithms. The choice of the algorithm depends on the application need, so it is difficult to always pick one winner. Influence of seeding density varies depending on what algorithm is used and where the seeding curve is placed. The increase in quality is associated with the extra computation cost and generation of higher number of mesh elements.

7.4 Conclusions

In this chapter, we presented four different stream surface verification methods to study various aspects of error involved in stream surface computation. While it is difficult to single out one specific algorithm that always gives the best stream surface in terms of quality, computation time and space utilization, the quad algorithm is the fastest and most space efficient, easy to implement and generally its stream surfaces are of good quality with some exceptions. We have also confirmed that the algorithm by Garth et al. can consistently produce good stream surfaces, albeit at higher computation cost. As an application of these metrics, users would be able to modify the parameter values of an algorithm by looking at the error amount and checking whether the error is decreased after the modification. As a next step, our proposed method can be extended to time-varying flow fields and streak surfaces, and more stream surface generation algorithms can be included for study. More knowledge about different stream surface generation algorithms can lead us to a new stream surface algorithm where we start the stream surface generation with a less complex algorithm and switch to a more accurate and time-consuming one when the error is increased.

Chapter 8: Conclusion and Future Work

8.1 Conclusions

This dissertation provides a detailed account of the uncertainty and error analysis techniques for scientific datasets. Specifically, we provide relationship uncertainty in the first two chapters. In these chapters, we focus on multivariate and ensemble datasets. For multivariate datasets, analysis of relationship is of prime interest to the scientists and we show how information theory can be used to design a system that can provide guidance to the users at every exploration stage. In this framework, we apply mutual information for generating an initial grouping of variables such that the variables with high information overlap can be placed in the same group. The importance of the variables within the sub-groups is identified by the calculation of conditional entropy. The variables are presented in the form of a force-directed graph layout to the users for interactive selection of variables. The selected variables are used to compute the specific mutual information to identify the scalar values of one variable which are informative about the other variables. For the exploration in the data domain, the visualization is performed using Parallel Coordinate Plots. For exploration in the spatial domain, isosurfaces are generated which are color mapped to other variables to show the degree of uncertainty about that variable.

Continuing with the relationship analysis, the following chapter outlines a system that is intended for exploring the sensitivity of the input parameters and accuracy of the output precipitation across resolutions of an ensemble dataset. In this chapter, we use a global sensitivity measure named δ to compute the sensitivity of the different input parameters over spatial regions and over temporal domain for the three given resolutions. To explore how the sensitivity of the parameters change over spatial regions, a location-based clustering is applied for different days and the clustering-agreement among the three resolutions is computed. For analyzing temporal patterns, spatially aggregated dataset is used for sensitivity computation given all the parameters and an MDS plot linked with line charts is presented for user interaction. For accuracy exploration, a Bayes rule based likelihood method is used to compute the local best predictor among the three resolutions for every spatial location from the ensembles. To create local probability density functions for this likelihood comparison, KDE approach is used. The local best predictors are then aggregated in the temporal domain to find which location is better predicted by which resolution over the total number of days and it is color-mapped for further user interaction. Users can now select regions from this image and analyze the temporal and spatial error pattern in a linked window. We carried out the experiments in close collaboration with an expert whose feedback establishes the efficacy of our proposed system.

Chapter 5 presents a novel vortex analysis framework that examines the uncertainty contained in four existing local vortex detection methods and combines them to perform the vortex analysis and detection in a more robust way. Since there are multiple vortex detectors available to us, we use these detectors as the inputs to our system and model the uncertainty of their predictions using a sigmoid function to convert the outputs of these detectors to a possibility value range of 0 and 1. These possibility values denote the certainty that a point

will be classified as a part of a vortex region. We apply a voting algorithm to classify the more certain vortex regions and cluster these regions based on their spatial locations. Next, we introduce the use of spatial proximity and classify the remaining points as vortex or non-vortex based on their distance from the closest vortex cluster. We worked closely with a domain expert and used the data marked by the expert to compare our method with other existing methods. We applied our method on multiple datasets and time steps and showed that the accuracy can be improved by systematically analyzing these flow structures.

Chapter 6 proposes a novel error-based technique for generating streamlines in parallel with high scalability for large two-dimensional flow fields. Instead of applying the traditional numerical integration schemes on the vector fields, we propose to compute a flux-based scalar field for implicit streamlines in parallel. Initially, data is subdivided into smaller blocks and local flux field is generated whose isocontours represent local streamlines. Since each block computes flux values according to a local flux origin, alignment of flux origins is needed to generate coherent streamlines across blocks. To achieve this, we use the additive property of flux and each block sends its local offset to its neighbors. Since this communication stage involves sending one float value across the neighbors of each block, it is efficiently performed and a stitched flux field is generated. Finally, isocontours of this stitched field are extracted in parallel to produce the final streamlines. The results obtained from a variety of datasets, both analytical and simulated, show the effectiveness of the stitching method. Our experiments also reveal good strong and weak scaling properties as well as much improved execution time.

The final chapter presents four different stream surface verification methods to study various aspects of error involved in stream surface computation. While it is difficult to single out one specific algorithm that always gives the best stream surface in terms of

quality, computation time and space utilization, the quad algorithm is the fastest and most space efficient, easy to implement and generally its stream surfaces are of good quality with some exceptions. We have also confirmed that the algorithm by Garth et al. can consistently produce good stream surfaces, albeit at higher computation cost. As an application of these metrics, users would be able to modify the parameter values of an algorithm by looking at the error amount and checking whether the error is decreased after the modification.

8.2 Future Work

This dissertation opens up many new research opportunities in the field of scientific visualization and data analysis. Since the supercomputers that generate simulation data are becoming more powerful and can produce even finer resolutions of data in space and time, quantification of uncertainty continues to be a challenge. For multivariate datasets, adapting our techniques to handle time-varying datasets remains a future work. Further, our system can be extended where ensemble multivariate datasets are involved. For ensemble input-output relationship analysis, our future work includes the use of multiple sensitivity analysis measures for more robustness. Also, we plan to provide an AMR grid based interactive cost optimization strategy for the output accuracy analysis. Since the datasets can be so large, we further plan to extend our system to *in situ* workflow and also to involve multiple domain experts for their feedback. For the uncertain vortex analysis, the immediate future work would be to reduce the reliance on the domain expert's labels for generating the results. Further, it requires research to develop a strategy to combine the labels from multiple experts and also automate the label generation process. Also, understanding the uncertain vortex-like features where the data itself contains uncertainty, remains another

important next step. Extending the error-based streamlines generation algorithm, it is important to make it work with three-dimensional flows and also compressible flows so that this method can be generalized. Further, in the presence of uncertainty stemming from the datasets itself, such a strategy will need to be devised. As the next step of stream surface error computation, we plan to extend that for time-varying flow fields and streak surfaces and include more stream surface generation algorithms for study. More knowledge about different stream surface generation algorithms can lead us to a new stream surface algorithm where we start the stream surface generation with a less complex algorithm and switch to a more accurate and time-consuming one when the error is increased.

Bibliography

[1] A. Agranovsky, D. Camp, C. Garth, E.W. Bethel, K.I. Joy, and H. Childs. Improved post hoc flow analysis via lagrangian representations. In *Large Data Analysis and Visualization*, pages 67–75, Nov 2014.

[2] Hiroshi Akiba, Kwan-Liu Ma, Jacqueline H. Chen, and Evatt R. Hawkes. Visualizing multivariate volume data from turbulent combustion simulations. *IEEE Computing in Science and Engineering*, pages 86–93, March/April 2007.

[3] M.F. Alam, D.K. Walters, and D. Thompson. Evaluation of a dynamic hybrid rans/les modeling methodology for attached and separated flows. *ASME Journal for Fluids Engineering (submitted)*.

[4] Helmut Alt, Peter Braß, Michael Godau, Christian Knauer, and Carola Wenk. Computing the Hausdorff distance of geometric patterns and shapes. In *Discrete and Computational Geometry. The Goodman–Pollack Festschrift*, volume 25, pages 65–76. 2003.

[5] A.O. Artero, M.C.F. de Oliveira, and H. Levkowitz. Enhanced high dimensional data visualization through dimension reduction and attribute arrangement. In *Information Visualization, 2006. IV 2006. Tenth International Conference on*, pages 707–712, 2006.

[6] Chandrajit L. Bajaj, Valerio Pascucci, and Daniel R. Schikore. The contour spectrum. In *Vis '97: Proceedings of the IEEE Visualization*, VIS '97, pages 167–173. IEEE Computer Society Press, 1997.

[7] D.C. Banks and B.a. Singer. A predictor-corrector technique for visualizing unsteady flow. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):151–163, June 1995.

[8] Steven B Beale. Visualisation of three-dimensional flow fields using two stream functions. 1997.

[9] K. Bensema, L. Gosink, H. Obermaier, and K. Joy. Modality-driven classification and visualization of ensemble variance. *IEEE Transactions on Visualization and Computer Graphics*, PP(99):1–1, 2015.

[10] R. Benzid, D. Arar, and M. Bentoumi. A fast technique for gray level image thresholding and quantization based on the entropy maximization. In *Systems, Signals and Devices, 2008. IEEE SSD 2008. 5th International Multi-Conference on*, pages 1–4, July 2008.

[11] H. Bhatia, S. Jadhav, P.-T. Bremer, Guoning Chen, J.A. Levine, L.G. Nonato, and V. Pascucci. Edge maps: Representing flow with bounded error. In *Pacific Visualization Symposium (PacificVis), 2011 IEEE*, pages 75–82, March 2011.

[12] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[13] A. Bock, A. Pembroke, M. Mays, L. Rastaetter, T. Ropinski, and A. Ynnerman. Visual verification of space weather ensemble simulations. In *2015 IEEE Scientific Visualization Conference*, pages 17–24, Oct 2015.

[14] U.D. Bordoloi and Han-Wei Shen. View selection for volume rendering. In *Visualization, 2005. VIS 05. IEEE*, pages 487–494, 2005.

[15] E. Borgonovo. A new uncertainty importance measure. *Reliability Engineering and System Safety*, 92(6):771–784, 2007.

[16] Neill E. Bowler, Alberto Arribas, Kenneth R. Mylne, Kelvyn B. Robertson, and Sarah E. Beare. The MOGREPS short-range ensemble prediction system. *Q.J.R. Meteorol. Soc.*, 134(632):703–722, apr 2008.

[17] R. Bramon, I. Boada, A. Bardera, J. Rodriguez, M. Feixas, J. Puig, and M. Sbert. Multimodal data fusion based on mutual information. *Visualization and Computer Graphics, IEEE Transactions on*, 18(9):1574 –1587, sept. 2012.

[18] Stefan Bruckner and Torsten Möller. Isosurface similarity maps. *Computer Graphics Forum*, 29:773–782, 2010.

[19] P.G. Buning. Sources of error in the graphical analysis of cfd results. *J. Sci. Comp*, 3, 1988.

[20] Raphael Bürger, Philipp Muigg, Martin Ilcik, Helmut Doleisch, and Helwig Hauser. Integrating local feature detectors in the interactive visual analysis of flow simulation data. In *Proceedings of Eurographics/ IEEE-VGTC Symposium on Visualization 2007*, pages 171–178, 2007.

[21] Dan G. Cacuci. Sensitivity theory for nonlinear systems. i. nonlinear functional analysis approach. *Journal of Mathematical Physics*, 22(12), 1981.

[22] David Camp, Christoph Garth, Hank Childs, David Pugmire, and Kenneth Joy. Streamline Integration using MPI-Hybrid Parallelism on Large Multi-Core Architecture. *Visualization and Computer Graphics, IEEE Transactions on*, (99):1–1, December 2010.

[23] Hamish Carr, Jack Snoeyink, and Ulrike Axen. Computing contour trees in all dimensions. *Computational Geometry*, 24(2):75 – 94, 2003.

[24] W. Castaings, D. Dartus, F.-X. Le Dimet, and G.-M. Saulnier. Sensitivity analysis and parameter estimation for distributed hydrological modeling: potential of variational methods. *Hydrology and Earth System Sciences*, 13(4):503–517, 2009.

[25] Pinaki Chakraborty, S. Balachandar, and Ronald J. Adrian. On the relationships between local vortex identification schemes. *Journal of Fluid Mechanics*, pages 189–214, 7 2005.

[26] M. Chen and H. Jänicke. An information-theoretic framework for visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 16(6):1206–1215, 2010.

[27] M. S. Chong, A. E. Perry, and B. J. Cantwell. A general classification of three-dimensional flow fields. *Physics of Fluids A: Fluid Dynamics*, 2:765–777, 1990.

[28] J.H.T. Claessen and J.J. van Wijk. Flexible linked axes for multivariate data visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2310–2316, 2011.

[29] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory 2nd Edition (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2006.

[30] T. F. Cox and M. A. A. Cox. *Multidimensional Scaling*. Chapman & Hall, London, 2001.

[31] F. Dambreville. Combining evidences by means of the entropy maximization principle. In *Information Fusion, 2007 10th International Conference on*, pages 1–8, July 2007.

[32] A. Dasgupta and R. Kosara. Pargnostics: Screen-space metrics for parallel coordinates. *Visualization and Computer Graphics, IEEE Transactions on*, 16(6):1017–1026, 2010.

[33] M.C.F. de Oliveira and H. Levkowitz. From visual data exploration to visual data mining: a survey. *Visualization and Computer Graphics, IEEE Transactions on*, 9(3):378–394, 2003.

[34] I. Demir, C. Dick, and R. Westermann. Multi-charts for comparative 3d ensemble visualization. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2694–2703, Dec 2014.

[35] Creon Levit Dennis C. Jespersen. Numerical Simulation of Flow Past a Tapered Cylinder. In *29th Aerospace Sciences Meeting*, 1991. AIAA Paper 91-0751.

[36] M. R. Deweese and M. Meister. How to measure the information gained from one symbol. *Network: Computation in Neural Systems*, (4):325–340, nov 1999.

[37] D.M. Driver and H.L. Seegmiller. Features of a reattaching turbulent shear layer in divergent channel flow. *AIAA journal*, 23:163–171, 1985.

[38] Yves Dubief and Franck Delcayre. On coherent-vortex identification in turbulence. *Journal of Turbulence*, page N11, 2000.

[39] Brian Duffy, Hamish Carr, and Torsten Möller. Integrating isosurface statistics and histograms. *IEEE Trans. Vis. Comput. Graph.*, 19(2):263–277, 2013.

[40] Matt Edmunds, Robert S. Laramee, Guoning Chen, Nelson Max, Eugene Zhang, and Colin Ware. Surface-based flow visualization. *Computers & Graphics*, 36:974–990, 2012.

[41] David G. Evans and Jeffrey P. Raffensperger. On the stream function for variable-density groundwater flow. *Water Resources Research*, 28(8):2141–2145, 1992.

[42] Miquel Feixas, Esteve Del Acebo, Philippe Bekaert, and Mateu Sbert. An information theory framework for the analysis of scene complexity, 1999.

[43] F. Ferstl, K. Brger, and R. Westermann. Streamline variability plots for characterizing the uncertainty in vector field ensembles. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):767–776, Jan 2016.

[44] C. Garth, H. Krishnan, X. Tricoche, T. Bobach, and K.I. Joy. Generation of accurate integral surfaces in time-dependent vector fields. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1404 –1411, 2008.

[45] Christoph Garth, Xavier Tricoche, Tobias Salzbrunn, Tom Bobach, and Gerik Scheuermann. Surface techniques for vortex visualization. In *VisSym 2004, Symposium on Visualization, Konstanz, Germany, May 19-21, 2004*, page 155, 2004.

[46] J.H. Giese. Stream functions for three-dimensional flows. *J. Math. Phys.*, Vol: 30, No. 1, Apr 1951.

[47] L. Gosink, K. Bensema, T. Pulsipher, H. Obermaier, M. Henry, H. Childs, and K. I. Joy. Characterizing and visualizing predictive uncertainty in numerical ensembles through bayesian model averaging. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2703–2712, Dec 2013.

[48] L.J. Gosink, C. Garth, J.C. Anderson, E.W. Bethel, and K.I. Joy. An application of multivariate statistical analysis for query-driven visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 17(3):264–275, 2011.

[49] Laurent Graftieaux, Marc Michard, and Nathalie Grosjean. Combining piv, pod and vortex identification algorithms for the study of unsteady turbulent swirling flows. *Measurement Science and Technology*, 2001.

[50] M. S. Greywall. Streamwise computation of three-dimensional flows using two stream functions. *Journal of Fluids Engineering*, 115:233–238, 1993.

[51] S. Gumhold. Maximum entropy light source placement. In *Visualization, 2002. VIS 2002. IEEE*, pages 275–282, 2002.

[52] H. Guo, X. Yuan, J. Huang, and X. Zhu. Coupled ensemble flow line advection and analysis. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2733–2742, Dec 2013.

[53] Hanqi Guo, He Xiao, and Xiaoru Yuan. Scalable multivariate volume visualization and analysis based on dimension projection and parallel coordinates. *Visualization and Computer Graphics, IEEE Transactions on*, 18(9):1397–1410, 2012.

[54] Zhun Guo, Minghuai Wang, Yun Qian, Vincent E. Larson, Steven Ghan, Mikhail Ovchinnikov, Peter A. Bogenschutz, Chun Zhao, Guang Lin, and Tianjun Zhou. A sensitivity analysis of cloud properties to clubb parameters in the single-column community atmosphere model (scam5). *Journal of Advances in Modeling Earth Systems*, 6(3):829–858, 2014.

[55] J. Hacker, S. Ha, C. Snyder, J. Berner, F. Eckel, E. Kuchera, M. Pocernich, S. Rugg, J. Schramm, and X. Wang. The u.s. air force weather agencys mesoscale ensemble: scientific description and performance results. *Tellus A*, 63(3), 2011.

[56] G. Haller. An objective definition of a vortex. *Journal of Fluid Mechanics*, pages 1–26, 2005.

[57] Lihua Hao, C. G. Healey, and S. A. Bass. Effective visualization of temporal ensembles. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):787–796, Jan 2016.

[58] T. Hollt, A. Magdy, Peng Zhan, Guoning Chen, G. Gopalakrishnan, I. Hoteit, C. D. Hansen, and M. Hadwiger. Ovis: A framework for visual analysisof ocean forecast ensembles. *IEEE Transactions on Visualization and Computer Graphics*, 20(8):1114–1126, Aug 2014.

[59] Z Hou, M Huang, LYR Leung, G Lin, and DM Ricciuto. Sensitivity of surface flux simulations to hydrologic parameters based on an uncertainty quantification framework applied to the community land model. *Journal of Geophysical Research. D. (Atmospheres)*, 117(D15):D15108, 2012.

[60] C Y Huang and G S Dulikravich. Stream function and stream-function-coordinate (sfc) formulation for inviscid flow field calculations. *Comput. Methods Appl. Mech. Eng.*, 59(2):155–177, nov 1986.

[61] J. P. M. Hultquist. Constructing stream surfaces in steady 3d vector fields. In *Proceedings of the 3rd conference on Visualization '92*, VIS '92, pages 171–178, Los Alamitos, CA, USA, 1992.

[62] J. C. R. Hunt, A. Way, and P. Moin. Eddies, stream, and convergence zones in turbulent flows. Center for Turbulence Research Report CTR-S88, Center for turbulence research, Standford University, 1988.

[63] C. B. Hurley and R. W. Oldford. Pairwise display of high-dimensional information via eulerian tours and hamiltonian decompositions. *Journal of Computational and Graphical Statistics*, 19(4):861–886, 2010.

[64] T. Hllt, A. Magdy, G. Chen, G. Gopalakrishnan, I. Hoteit, C. D. Hansen, and M. Hadwiger. Visual analysis of uncertainties in ocean forecasts for planning and operation of off-shore structures. In *Visualization Symposium (PacificVis), 2013 IEEE Pacific*, pages 185–192, Feb 2013.

[65] Richard Ibbitt and Terence ODonnell. Sensitivity theory for nonlinear systems. i. nonlinear functional analysis approach. *IAHS Publication*, 101:462–475, 1971.

[66] A. Inselberg. Multidimensional detective. In *Information Visualization, 1997. Proceedings., IEEE Symposium on*, pages 100–107, 1997.

[67] A. Inselberg and Bernard Dimsdale. Parallel coordinates: a tool for visualizing multi-dimensional geometry. In *Visualization, 1990. Visualization '90., Proceedings of the First IEEE Conference on*, pages 361–378, 1990.

[68] Alfred Inselberg. The plane with parallel coordinates. *The Visual Computer*, 1(2):69–91, August 1985.

[69] Bertrand Iooss and Paul Lemaître. *Uncertainty Management in Simulation-Optimization of Complex Systems: Algorithms and Applications*, chapter A Review on Global Sensitivity Analysis Methods, pages 101–122. Springer US, 2015.

[70] H. Jänicke, M. Bottinger, and G. Scheuermann. Brushing of attribute clouds for the visualization of multivariate data. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1459–1466, 2008.

[71] M. Jarema, I. Demir, J. Kehrer, and R. Westermann. Comparative visual analysis of vector field ensembles. In *Visual Analytics Science and Technology (VAST), 2015 IEEE Conference on*, pages 81–88, Oct 2015.

[72] Jinhee Jeong and Fazle Hussain. On the identification of a vortex. *Journal of Fluid Mechanics*, 285:69–94, 1 1995.

[73] Ming Jiang, Raghu Machiraju, and David Thompson. Geometric verification of swirling features in flow fields. In *Proceedings of the Conference on Visualization '02*, VIS '02, pages 307–314, 2002.

[74] Ming Jiang, Raghu Machiraju, and David Thompson. Detection and visualization of vortices. In *The Visualization Handbook*, pages 295–309. Academic Press, 2005.

[75] J. Johansson and M. Cooper. A screen space quality method for data abstraction. *Computer Graphics Forum*, 27(3):1039–1046, 2008.

[76] P. R. Johnston and D. H. Pilgrim. Parameter optimization for watershed models. *Water Resources Research*, 12(3):477–486, 1976.

[77] JakobJ. Keller. A pair of stream functions for three-dimensional vortex flows. *Zeitschrift fr angewandte Mathematik und Physik ZAMP*, 47(6):821–836, 1996.

[78] Wesley Kendall, Jingyuan Wang, Melissa Allen, Tom Peterka, Jian Huang, and David Erickson. Simplified parallel domain traversal. *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis on - SC '11*, page 1, 2011.

[79] D.N. Kenwright and G.D. Mallinson. A 3-d streamline tracking algorithm using dual stream functions. In *Visualization, 1992. Visualization '92, Proceedings., IEEE Conference on*, pages 62–68, Oct 1992.

[80] M. Khoury and R. Wenger. On the fractal dimension of isosurfaces. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1198 –1205, 2010.

[81] D. Knight and G. Mallinson. Visualizing unstructured flow data using dual stream functions. *Visualization and Computer Graphics, IEEE Transactions on*, 2(4):355–363, Dec 1996.

[82] B. Kohler, R. Gasteiger, U. Preim, H. Theisel, M. Gutberlet, and B. Preim. Semi-automatic vortex extraction in 4d pc-mri cardiac blood flow data using line predicates. *IEEE Transactions on Visualization and Computer Graphics*, pages 2773–2782, 2013.

[83] Robert S. Laramee, Christoph Garth, Jürgen Schneider, and Helwig Hauser. Texture advection on stream surfaces: a novel hybrid visualization applied to cfd simulation results. In *Proceedings of the Eighth Joint Eurographics / IEEE VGTC conference on Visualization*, EUROVIS'06, pages 155–162, 2006.

[84] Faming Liang, Yichen Cheng, and Guang Lin. Simulated stochastic approximation annealing for global optimization with a square-root cooling schedule. *Journal of the American Statistical Association*, 109(506):847–863, 2014.

[85] Liang Fu Lu, Mao Lin Huang, and Tze-Haw Huang. A new axes re-ordering method in parallel coordinates visualization. In *Machine Learning and Applications (ICMLA), 2012 11thInternational Conference on*, volume 2, pages 252–257, 2012.

[86] W. J. Conover M. D. McKay, R. J. Beckman. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979.

[87] A.R. Martin and M.O. Ward. High dimensional brushing for interactive exploration of multivariate data. In *Visualization, 1995. Visualization '95. Proceedings., IEEE Conference on*, pages 271–, 1995.

[88] George B. Matanga. Stream functions in three-dimensional groundwater flow. *Water Resources Research*, 29(9):3125–3133, 1993.

[89] Tony McLoughlin, Robert S. Laramee, Ronald Peikert, Frits H. Post, and Min Chen. Over two decades of integration-based, geometric flow visualization. *Computer Graphics Forum*, 29(6):1807–1829, 2010.

[90] Tony McLoughlin, Robert S. Laramee, and Eugene Zhang. Easy integral surfaces: a fast, quad-based stream and path surface algorithm. In *Proceedings of the 2009 Computer Graphics International Conference*, CGI '09, pages 73–82, New York, NY, USA, 2009.

[91] M. Mirzargar, R. T. Whitaker, and R. M. Kirby. Curve boxplot: Generalization of boxplot for ensembles of curves. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2654–2663, Dec 2014.

[92] Max D. Morris. Factorial sampling plans for preliminary computational experiments. *Technometrics*, 33(2):161–174, 1991.

[93] Cornelius Muller, David Camp, Bernd Hentschel, and Christoph Garth. Distributed parallel particle advection using work requesting. In *IEEE Symposium on Large Data Analysis and Visualization 2013, LDAV 2013 - Proceedings*, pages 1–6, 2013.

[94] E. Murman and K. Powell. Trajectory integration in vortical flows. *AIAA Journa*, 27:982–984, 1988.

[95] Boonthanome Nouanesengsy, Teng-Yok Lee, and Han-Wei Shen. Load-balanced parallel streamline generation on large scale vector fields. *IEEE transactions on visualization and computer graphics*, 17(12):1785–94, December 2011.

[96] Mathias Otto and Holger Theisel. Vortex analysis in uncertain vector fields. *Computer Graphics Forum*, 31(3pt2):1035–1044, 2012.

[97] H.-G. Pagendarm, B. Henne, and M. Rutten. Detecting vortical phenomena in vector data by medium-scale correlation. In *Visualization '99. Proceedings*, pages 409–552, Oct 1999.

[98] Alex T. Pang, Craig M. Wittenbrink, and Suresh K. Lodha. Approaches to uncertainty visualization. *The Visual Computer*, 13:370–390, 1997.

[99] Ronald Peikert and Martin Roth. The "parallel vectors" operator - a vector field visualization primitive. In *IEEE Visualization*, pages 263–270, 1999.

[100] W. Peng, M.O. Ward, and E.A. Rundensteiner. Clutter reduction in multi-dimensional data visualization using dimension reordering. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pages 89–96, 2004.

[101] A. E. Perry, M. S. Chong, and T. T. Lim. The vortex-shedding process behind two-dimensional bluff bodies. *Journal of Fluid Mechanics*, 116:77–90, 3 1982.

[102] Tom Peterka and R Ross. A study of parallel particle tracing for steady-state and time-varying flow fields. *Parallel & Distributed Processing Symposium (IPDPS), 2011 IEEE International*, pages 580–591, 2011.

[103] Konrad Polthier and Eike Preuß. *Visualization and Mathematics III*, chapter Identifying Vector Field Singularities Using a Discrete Hodge Decomposition, pages 113–134. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[104] Kai Pothkow and Hans-Christian Hege. Positional uncertainty of isocontours: Condition analysis and probabilistic measures. *IEEE Transactions on Visualization and Computer Graphics*, 17(10):1393–1406, 2011.

[105] Kristin Potter, Andrew Wilson, Peer-Timo Bremer, Dean Williams, Charles Doutriaux, Valerio Pascucci, and Chris Johhson. Visualization of uncertainty and ensemble data: Exploration of climate modeling and weather forecast data with integrated visus-cdat systems. *Journal of Physics: Conference Series*, 180(1):012089, 2009.

[106] Kai Pthkow and Hans-Christian Hege. Nonparametric models for uncertainty visualization. *Computer Graphics Forum*, 32(3pt2):131–140, 2013.

[107] S K Robinson. Coherent motions in the turbulent boundary layer. *Annual Review of Fluid Mechanics*, 23(1):601–639, 1991.

[108] Martin Roth. *Automatic Extraction of Vortex Core Lines and Other Line-Type Features for Scientific Visualization*. PhD Dissertation No. 13673, ETH Zurich, 2000. published by Hartung-Gorre Verlag, Konstanz, ISBN 3-89649-582-8.

[109] Martin Roth and Ronald Peikert. A Higher-Order Method for Finding Vortex Core Lines. In *Proceedings IEEE Visualization 1998*, pages 143–150, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press.

[110] O. Rubel, Prabhat, Kesheng Wu, H. Childs, J. Meredith, C. G R Geddes, E. Cormier-Michel, S. Ahern, G.H. Weber, P. Messmer, H. Hagen, B. Hamann, and E.W. Bethel. High performance multivariate visual data exploration for extremely large data. In *High Performance Computing, Networking, Storage and Analysis, 2008. SC 2008. International Conference for*, pages 1–12, 2008.

[111] I. Ari Sadarjoen, Frits H. Post, Bing Ma, David C. Banks, and Hans-Georg Pagendarm. Selective visualization of vortices in hydrodynamic flows. In *IEEE Visualization*, pages 419–422, 1998.

[112] A. Saltelli, S. Tarantola, and K. P.-S. Chan. A quantitative model-independent method for global sensitivity analysis of model output. *Technometrics*, 41(1):39–56, 1999.

[113] Andrea Saltelli. Sensitivity analysis for importance assessment. *Risk Analysis*, 22(3):579–590, 2002.

[114] J. Sanyal, S. Zhang, J. Dyer, A. Mercer, P. Amburn, and R. Moorhead. Noodles: A tool for visualization of numerical weather model ensemble uncertainty. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1421–1430, Nov 2010.

[115] Tobias Schafhitzel, Eduardo Tejada, Daniel Weiskopf, and Thomas Ertl. Point-based stream surfaces and path surfaces. In *Proceedings of Graphics Interface 2007*, GI '07, pages 289–296, New York, NY, USA, 2007.

[116] Tobias Schafhitzel, Joachim E. Vollrath, Joao Paulo Gois, Daniel Weiskopf, Antonio Castelo, and Thomas Ertl. Topology-preserving lambda$_2$-based vortex core line detection for flow visualization. *Comput. Graph. Forum*, 2008.

[117] C.E. Scheidegger, J.M. Schreiner, B. Duffy, H. Carr, and C.T. Silva. Revisiting histograms and isosurface statistics. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1659 –1666, 2008.

[118] Gerik Scheuermann, Tom Bobach, Hans Hagen, Karim Mahrous, Bernd Hamann, Kenneth I. Joy, and Wolfgang Kollmann. A tetrahedra-based stream surface algorithm. In *Proceedings of IEEE Visualization 2001*, pages 83–91, 2001.

[119] Dominic Schneider, Wieland Reich, Alexander Wiebel, and Gerik Scheuermann. Topology aware stream surfaces. *Comput. Graph. Forum*, 29(3):1153–1161, 2010.

[120] William J. Schroeder, Lisa S. Avila, and William Hoffman. Visualizing with vtk: A tutorial. *IEEE Comput. Graph. Appl.*, 20(5):20–27, September 2000.

[121] M. Schulze, T. Germer, C. Rössl, and H. Theisel. Stream surface parametrization by flow-orthogonal front lines. *Computer Graphics Forum (Proc. SGP)*, 31(5):1725–1734, 2012.

[122] Rainer K. Senger and Graham E. Fogg. Stream functions and equivalent freshwater heads for modeling regional flow of variable-density groundwater: 2. application and implications for modeling strategy. *Water Resources Research*, 26(9):2097–2106, 1990.

[123] A. Sherif and M. Hafez. Computation of three-dimensional transonic flows using two stream functions. *International Journal for Numerical Methods in Fluids*, 8(1):17–29, 1988.

[124] Jeremy Siek, Lie-Quan Lee, and Andrew Lumsdaine. Boost graph library. http://www.boost.org/libs/graph/, June 2000.

[125] Bernard W Silverman, editor. *Density estimation for statistics and data analysis*. Chapman & Hall/CRC, 1992.

[126] I.M. Sobol and S. Kucherenko. Derivative based global sensitivity measures and their link with global sensitivity indices. *Mathematics and Computers in Simulation*, 79(10):3009 – 3017, 2009.

[127] I.M Sobol. Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. *Mathematics and Computers in Simulation*, 55(13):271 – 280, 2001.

[128] Simon Stegmaier, Ulrich Rist, and Thomas Ertl. Opening the can of worms: An exploration tool for vortical flows. In *IEEE Visualization*, page 59. IEEE Computer Society, 2005.

[129] T. Stöter, T. Weinkauf, H.-P. Seidel, and H. Theisel. Implicit integral surfaces. In *Proc. Vision, Modeling and Visualization*, page to appear, Magdeburg, Germany, November 2012.

[130] David S. Thompson, Jaya Sreevalsan Nair, Satya Sridhar Dusi Venkata, Raghu K. Machiraju, Ming Jiang, and Gheorghe Craciun. Physics-based feature mining for large data exploration. *Computing in Science and Engg.*, 4(4):22–30, 2002.

[131] Yiying Tong, Santiago Lombeyda, Anil N. Hirani, and Mathieu Desbrun. Discrete multiscale vector field decomposition. In *ACM SIGGRAPH 2003 Papers*, SIGGRAPH '03, pages 445–452, New York, NY, USA, 2003. ACM.

[132] Xavier Tricoche, Christoph Garth, Gordon L. Kindlmann, Eduard Deines, Gerik Scheuermann, Markus Rtten, and Charles D. Hansen. Visualization of intricate flow structures for vortex breakdown analysis. In *IEEE Visualization*, pages 187–194. IEEE Computer Society, 2004.

[133] Greg Turk and David Banks. Image-guided streamline placement, 1996. http://www.cc.gatech.edu/ turk/streamlines/streamlines.html.

[134] M van Lier-Walqui, T Vukicevic, and D J Posselt. Quantification of cloud microphysical parameterization uncertainty using radar reflectivity. *Mon Weather Rev*, 140, 2012.

[135] Jarke J. van Wijk. Implicit stream surfaces. In *Proceedings of the 4th Conference on Visualization '93*, VIS '93, pages 245–252, Washington, DC, USA, 1993. IEEE Computer Society.

[136] Pere-Pau Vázquez, Miquel Feixas, Mateu Sbert, and Wolfgang Heidrich. Automatic View Selection Using Viewpoint Entropy and its Application to Image-Based Modelling. *Computer Graphics Forum*, 22(4):689–700, 2003.

[137] I. Viola, M. Feixas, M. Sbert, and M.E. Groller. Importance-driven focus of attention. *Visualization and Computer Graphics, IEEE Transactions on*, 12(5):933–940, 2006.

[138] Chaoli Wang, Hongfeng Yu, R.W. Grout, Kwan-Liu Ma, and J.H. Chen. Analyzing information transfer in time-varying multivariate data. In *Pacific Visualization Symposium (PacificVis), 2011 IEEE*, pages 99–106, 2011.

[139] Chaoli Wang, Hongfeng Yu, and Kwan-Liu Ma. Importance-driven time-varying data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1547–1554, nov 2008.

[140] G Warren, S and H Schneider S. Seasonal simulation as a test for uncertainties in the parameterizations of a budyko-sellers zonal climate model. *J Atmos Sci*, 36:1377–1391, 1979.

[141] T. Weinkauf, J. Sahner, H. Theisel, and H.-C. Hege. Cores of swirling particle motion in unsteady flows. *Visualization and Computer Graphics, IEEE Transactions on*, pages 1759–1766, Nov 2007.

[142] D. Whalen and M. L. Norman. Competition data set and description. 2008 IEEE Visualization Design Contest, http://vis.computer.org/VisWeek2008/vis/contests.html, 2008.

[143] R. T. Whitaker, M. Mirzargar, and R. M. Kirby. Contour boxplots: A method for characterizing uncertainty in feature sets from simulation ensembles. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2713–2722, Dec 2013.

[144] Pak Chung Wong and R. Daniel Bergeron. 30 years of multidimensional multivariate visualization. In *Scientific Visualization, Overviews, Methodologies, and Techniques*, pages 3–33, Washington, DC, USA, 1997. IEEE Computer Society.

[145] J. Z. Wu, A. K. Xiong, and Y. T. Yang. Axial stretching and vortex definition. *Physics of Fluids 17*, 2005.

[146] Yang Xiang, David Fuhry, Ruoming Jin, Ye Zhao, and Kun Huang. Visualizing clusters in parallel coordinates for visual knowledge discovery. In *Proceedings of the 16th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining - Volume Part I*, PAKDD'12, pages 505–516, 2012.

[147] Lijie Xu, Teng-Yok Lee, and Han-Wei Shen. An information-theoretic framework for flow visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 16(6):1216–1224, 2010.

[148] H Yan, H Qian, Guang Lin, L Leung, B Yang, and Q Fu. Parametric sensitivity and calibration for the kain-fritsch convective parameterization scheme in the wrf model. *Climate Research*, 59:135–147, 2014.

[149] B. Yang, Y. Qian, G. Lin, R. Leung, and Y. Zhang. Some issues in uncertainty quantification and parameter tuning: a case study of convective parameterization scheme in the wrf regional climate model. *Atmospheric Chemistry and Physics*, 12(5):2409–2427, 2012.

[150] Di Yang, E.A. Rundensteiner, and M.O. Ward. Analysis guided visual exploration of multivariate data. In *Visual Analytics Science and Technology, 2007. VAST 2007. IEEE Symposium on*, pages 83–90, 2007.

[151] P. k. Yeung and S.B. Pope. An algorithm for tracking fluid particles in numerical simulations of homogeneous turbulence. *J. Comp. Physics.*, 79:373, 1988.

[152] C. S. Yih. Stream functions in three-dimensional flows. *La Houille Blanche*, (3):445–450, 1957.

[153] L. Zhang, Q. Deng, R. Machiraju, A. Rangarajan, D. Thompson, D. K. Walters, and H.-W. Shen. Boosting techniques for physics-based vortex detection. *Computer Graphics Forum*, 33(1):282–293, 2014.

[154] C. Zhao, X. Liu, Y. Qian, J. Yoon, Z. Hou, G. Lin, S. McFarlane, H. Wang, B. Yang, P.-L. Ma, H. Yan, and J. Bao. A sensitivity study of radiative fluxes at the top of atmosphere to cloud-microphysics and aerosol parameters in the community atmosphere model cam5. *Atmospheric Chemistry and Physics*, 13(21):10969–10987, 2013.

[155] Jianlong Zhou and Masahiro Takatsuka. Automatic transfer function generation using contour tree controlled residue flow model and color harmonics. *IEEE Transactions on Visualization and Computer Graphics*, 15:1481–1488, 2009.