Coupled Ensemble Flow Line Advection and Analysis

Hanqi Guo, *Student Member, IEEE*, Xiaoru Yuan, *Member, IEEE*, Jian Huang, *Member, IEEE*, and Xiaomin Zhu

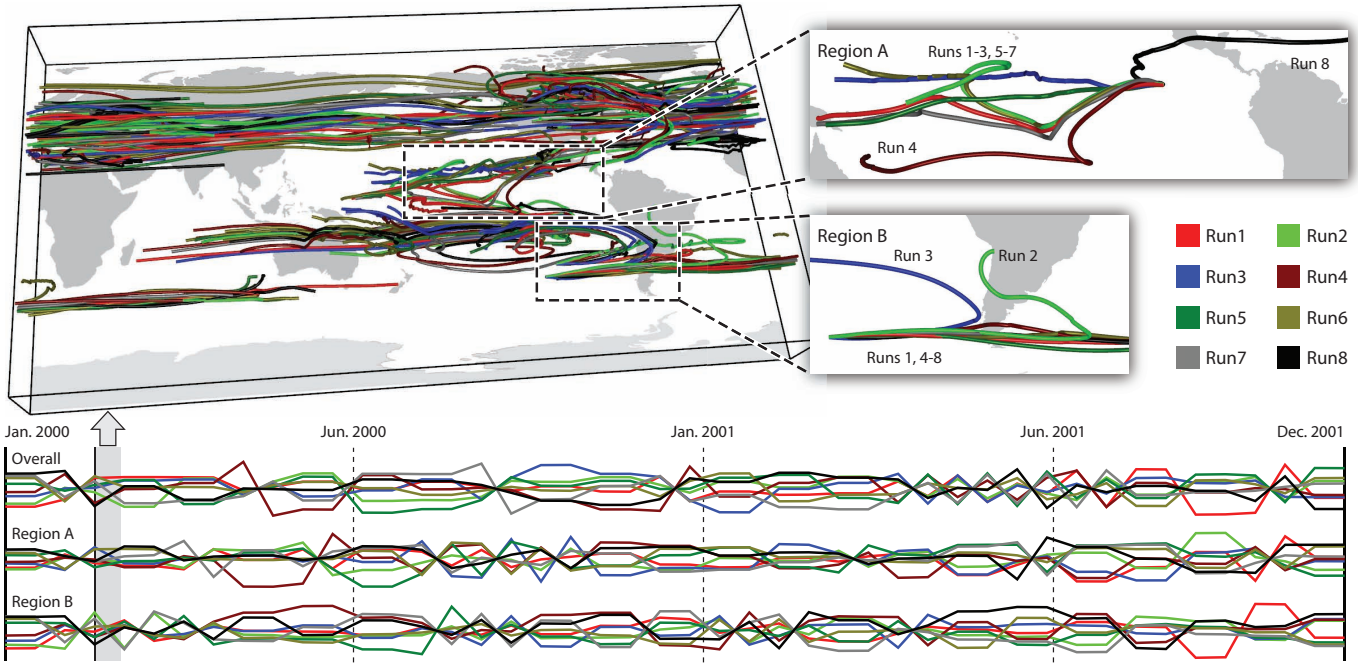

Fig. 1. Visualization results of GEOS-5 ensemble simulation data. Pathlines are filtered and selected with Lagrangian-based variation. The depiction summarizes the variation among pathlines between runs in the chosen time period. In region A, lines traced from the same point in different runs go in different directions, including Australia, west Pacific Ocean, and Caribbean Sea. Highly varying results are also in region B. In the timeline view the differences are presented in the whole volume as well as in regions A and B.

Abstract—Ensemble run simulations are becoming increasingly widespread. In this work, we couple particle advection with pathline analysis to visualize and reveal the differences among the flow fields of ensemble runs. Our method first constructs a variation field using a Lagrangian-based distance metric. The variation field characterizes the variation between vector fields of the ensemble runs, by extracting and visualizing the variation of pathlines within ensemble. Parallelism in a MapReduce style is leveraged to handle data processing and computing at scale. Using our prototype system, we demonstrate how scientists can effectively explore and investigate differences within ensemble simulations.

Index Terms—Ensemble analysis, parallel processing, field line advection

1 Introduction

In scientific applications, it is increasingly widespread to study model uncertainties and parameter sensitivities using ensemble runs. For example, climate researchers routinely use ensemble runs to reveal how greenhouse gas spreads over the earth under different physical, human and policy scenarios. Scientists also commonly conduct ensemble runs with different initial values and boundary conditions. Because a data ensemble can be large and complex, it is still an open problem to effectively extract and compare features across an ensemble.

In this work we focus on flow fields because they are prevalent in simulations. Wind fields in climate simulation, diffusion flux in pollution simulation, and advections in combustion simulation are a few well-known examples. Moreover, flow components are usually related to the core of the domain science at hand. While there are many methods for visualizing scalar variables in an ensemble or multi-run simulation, far fewer exist for studying flow fields in ensembles. It is difficult to extract flow field differences between runs, because unsteady flow features are sensitive to turbulences and not spatially localized. Previous work has been done to make visual comparisons [40], yet the field has not yet attempted to comprehensively analyze ensemble uncertainty in unsteady flow fields.

To study unsteady flow features in an ensemble as a whole, we have developed a novel framework, and a prototype system, called *e*FLAA (ensemble Flow Line Advection and Analysis). *e*FLAA computes a comprehensive variation field, which is then used for a variety of purposes, such as to better guide the analysis, to reduce the runtime, and to accelerate the analysis. *e*FLAA achieves scalability by extending

- *Hanqi Guo is with Key Laboratory of Machine Perception (Ministry of Education), School of EECS, and Center for Computational Science and Engineering, Peking University. E-mail: hanqi.guo@pku.edu.cn.*
- *Xiaoru Yuan is with Key Laboratory of Machine Perception (Ministry of Education), School of EECS, and Center for Computational Science and Engineering, Peking University. E-mail: xiaoru.yuan@pku.edu.cn.*
- *Jian Huang is with Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville. E-mail: huangj@eecs.utk.edu.*
- *Xiaomin Zhu is with National Super Computing Center in Jinan, Shandong, China. E-mail: zhuxm@sdas.org.*

Manuscript received 31 March 2013; accepted 1 August 2013; posted online 13 October 2013; mailed on 4 October 2013.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

DStep [18] in significant ways. First, field lines are massively traced across the runs in a synchronized manner. Second, tasks are scheduled so that memory use by intermediate field lines is limited while maintaining both load balance and high throughput.

One way to compare ensemble flow fields is to compute field line traces, store and compare them. Unfortunately, the volume of all these traces could be much larger than the original ensemble dataset. Another way is to compare field line traces while the traces are being computed, omitting a vast majority of the I/O overhead. We choose this latter way. The challenge and our focus is to achieve for the ensemble analysis scalability in terms of memory use, while balancing the scalability in terms of computation. The correctness of *e*FLAA is verified with a synthetic dataset, and then validated for effectiveness using two real world scientific datasets from atmospheric modeling of CO_2 transportation and from models for weather prediction. The system is evaluated on two different supercomputer architectures.

*e*FLAA is the first to employ a distance metric based on Lagrangian specification. Specifically, the scalar variation field, based on a Lagrangian specification represents the differences either between the geometry of pathlines or between physical quantities along such lines and thus reveals the ensemble uncertainty. With the flexibly-defined Lagrangian-based metric, differences of field lines between runs are compared on their original positions as well as advected positions. Based on the variation field, field lines can be selected to illustrate the ensemble differences between runs.

*e*FLAA provides elaborate and interactive user interfaces. After processing the data, the distances between lines originating from the same spatiotemporal locations are recorded in variation fields. Field lines with high variation values are also stored and rendered through the user interface. In the timeline view, users can navigate the results with the aid of similarity overview graphs.

In the following, we describe the background of our work in Section 2, illustrate conceptual design of *e*FLAA in Section 3, describe details of parallel algorithms and performance in Section 4, demonstrate application results in Section 5, and finally provide discussions and conclusions in Section 6 and Section 7, respectively.

2 Background

2.1 Ensemble Data Analysis, Comparative Visualization, and Uncertainty Visualization

Ensemble runs are commonly used for studying sensitivities of parameters, mitigating uncertainty and improving models. Ensemble data visualization research includes the visualization of uncertainty (e.g. means and standard deviations), comparative display, and user interfaces for navigation [34]. Although few previous work has been on visualizing flow variation in ensembles, our work is related to both comparative visualization and uncertainty visualization methods.

Comparative visualization aims at showing similarities and differences in the datasets [26]. Known techniques range from juxtaposition, superposition, to explicit representations [11]. Those techniques in general focus on visual representation and rendering, and should be considered under the framework of an analysis workflow [34]. Specific to ensemble data, comparative visualization methods include pseudo coloring, contour lines [22], and glyphs that encode scalar differences and outliers [15]. Verma and Pang [40] proposed several methods for making visual comparisons of flow fields datasets.

Uncertainty visualization relies upon the uncertainty information associated with ensemble data. Previous works often measure uncertainties by local statistics, such as mean value, standard deviations, and confidence intervals [34, 37]. Ensemble uncertainties are visualized in both attribute spaces [32] and in real space using metaphors such as uncertainty glyphs and ribbons [37]. Previous works on flow fields primarily use Eulerian specification of the ensemble flow fields. This does not suffice for revealing differences between particle trajectories. To address this shortcoming, our work uses Lagrangian-specification.

Beyond the scope of ensemble flow fields, there has been a lot of research in uncertainty quantification and visualization, either in general scientific visualization [13] or specifically in spatiotemporal data visualization [14, 31, 30]. As categorized by Pang et al. [27], there exists a wide range of methods(e.g. glyphs, texture mapping, animation) and a large variety of visual metaphors. Potter et al. [33] provides a more recent review on this topic as well. Independent of ensembles, uncertainty visualization has also attracted attention in the context of managing simulation processes [39], understanding parameter spaces [3], and differentiating simulation results [5]. While we draw inspiration from the above literature, our work is not directly related to these studies.

There are other noteworthy works on visualizing the uncertainty of flow fields independent of ensemble data. For example, visual metaphors such as glyphs [41] and textures [4] are effective at showing flow uncertainties. In UFLOW [21], uncertainties raised by different numerical particle tracing methods have been visualized. Flow Radar Graph [12] provides a glyph-based visualization for unsteady flow, which presents the changes in flow directions as spherical coordinates. Recently, techniques were proposed to visualize the uncertain topology [25] and local features [29] of 3D vector fields. Our research on ensemble flow field focuses more on characterizing the variation among ensemble flow fields, as opposed to the optimal visual representation of uncertainty.

In a related manner, for data assimilation in ensemble forecasting, climate modelers have developed Ensemble Kalman Filters [9]. Our goal is to reveal differences among field lines. Although these variations we reveal will be useful for making better ensemble forecasting, making predictions is beyond the scope of this work.

2.2 Parallel Field Line Advection

Scalable algorithms for tracing field lines are hard. As simulations surpass terascale and soon petascale, it has become unrealistic to extract a meaningful set of field lines without using supercomputers. To analyze differences in ensembles of flow fields, even more field lines have to be extracted and compared for corresponding locations in the ensemble runs. Since the data volume of the traced field lines exceeds that of the ensemble data, scalability is difficult to achieve with standard memory sizes with existing methods.

Tracing field lines can be parallelized using either data-parallel or task-parallel methods or both. Scalability is limited by I/O overhead and complex load balancing. Parallel flow visualization methods that are solely data-parallel rely on data block distribution for load-balancing. Known schemes range from static round-robin [28], to hierarchical clustering [43], to partitioning by flow directions and features [8]. Controlled block layout has also been leveraged for improving I/O performance [7]. Methods that are solely task parallel typically revolve around scheduling: such as dynamic load balancing [35], workload estimation [24], and on-demand strategies to reduce communication and I/O costs [6]. A dynamic load-balancing approach proposed by Pugmire et al. [35], which shows good scalability and performance. Different strategies for block I/O are applied on-demand to improve the throughput of the system. Camp et al. [6] presented a hybrid-MPI implementation for parallel streamline integration. The hybrid algorithm improves the overall performance by reducing the communication and I/O cost. A workload estimation algorithm is proposed to decompose the flow field statically in recent literature [24].

Recently, Kendall et al. [18] proposed a MapReduce-like framework *DStep* for field line tracing, Unlike data-parallel MapReduce, *DStep* which implements and successfully manages both data-parallel and task-parallel parallelism. *DStep* was reported to scale to 64K BlueGene/P cores [18]. For comparing the variation of pathlines among ensemble runs, the ability to extract many field lines at once is so crucial that it demands the most efficient and scalable solution. For this need, *DStep* is an ideal method. However, higher extraction performance increases the data deluge and precludes storing the entire field line dataset for later analysis. This is the same rationale that has motivated the recently popular topic of in-situ visualization. *DStep* was not originally designed for such a use-case: the key missing element is to have run-time control of how much memory is used and accordingly manage the "run-time flow" of concurrent tasks. *e*FLAA provides this

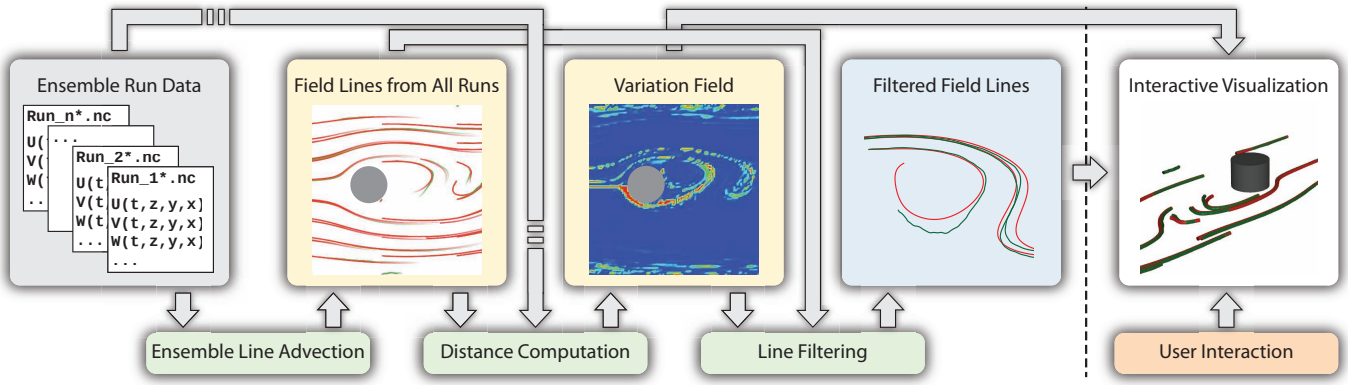

Fig. 2. Our system workflow. Field lines are traced and then used to compute variation and produce a variation field. Field lines are further filtered and selected according to the uncertainty variation field for users to investigate differences among the ensemble runs.

missing component.

Tracing massive amounts of field lines is also done in flow analysis like FTLE (Finite-Time Lyapunov Exponent). One way to accelerate FTLE is by reducing the number of particles [10]. Recently, a parallel framework was also proposed [23]. In comparison to FTLE computation, our problem requires consideration of larger spatial and temporal domains and thus incurs more stringent needs to limit memory use.

In Section 4 we further analyze the design needs and describe how *e*FLAA reimplements *DStep* and how we redesigned the system, including architectural extensions that improve scalability.

3 Variation Quantification for Ensemble Flows

The main purpose of our method is to discover the differences between multiple flow fields from ensemble runs. Essentially, we extract differences as features from flow fields.

Fig. 2 shows our workflow. The input is the raw data and the output is a set of field lines capable of depicting the differences. From the raw flow field data, we compute the distances between the runs at every spatiotemporal sample using the Lagrangian-based distance metric. The distance metric is computed by accumulating differences along the field lines that are traced from the samples. The variation field, which describes the variation of every spatiotemporal location, is obtained by averaging the distance fields. Then we filter out the samples with high variation, and visualize the field lines traced from the corresponding samples. A 2D cylinder unsteady flow simulation dataset is used to illustrate the proposed methods in Figure 3. A "noise" run is synthesized by adding Gaussian noises.

Since for real data the computation costs, memory footprint, and I/O costs are extremely high, careful design of a scalable and parallel system is needed.

3.1 Specification

In fluid dynamics, there are two kinds of specifications for the flow field, Eulerian and Lagrangian.

In the Eulerian specification, the variables are described as a function of spatiotemporal coordinate $\mathbf{x}$ and t. For example, the velocity field $\mathbf{v}$, the pressure field p, and the temperature field T can be written as the following, respectively:

$$\mathbf{v} = \mathbf{v}(\mathbf{x},t), p = p(\mathbf{x},t), T = T(\mathbf{x},t). \tag{1}$$

In Lagrangian specification, quantities are associated with particles that move within in the flow field. The particles are identified by their location $\mathbf{a}$ at time $t = 0$. For instance, the displacement $\mathbf{X}$, the pressure field p, and the temperature field T in Lagrangian specification are the following, respectively:

$$\mathbf{X} = \mathbf{X}(\mathbf{a},t), p = p(\mathbf{a},t), T = T(\mathbf{a},t). \tag{2}$$

The two specifications are two ways of describing the same phenomena, they are related as:

$$\mathbf{v}(\mathbf{X}(\mathbf{a},t),t) = \frac{\partial \mathbf{X}(\mathbf{a},t)}{\partial t}, \tag{3}$$

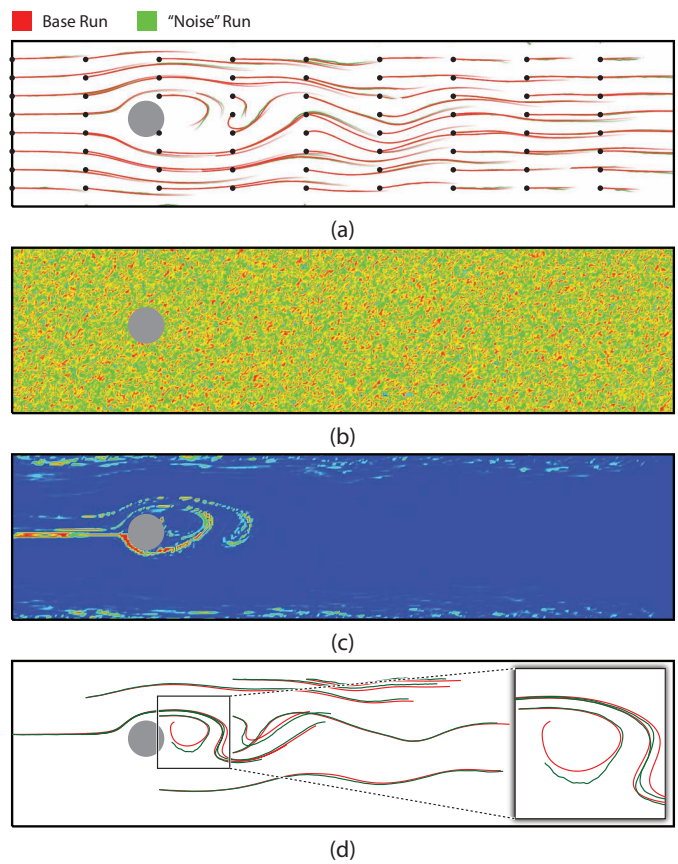

Fig. 3. The cylinder dataset (Reynolds number is 100) on a 400×100 regular grid. Gaussian noise ($\mu = 0, \sigma^2 = 0.04$) has been added to each velocity component. (a) The ensemble pathline advection results. The lines are intentionally faded over time for illustration; (b) The vertex-wise differences between the two runs, namely the Eulerian-based differences; (c) The Lagrangian-based differences of the two runs, which is based on the ensemble pathline advection results; (d) The filtered pathlines which can characterize the differences between the two runs.

$\mathbf{X}(\mathbf{a},t)$ is a pathline generated from the spatiotemporal coordinates $(\mathbf{a},0)$. In general, for any quantities in the flow field, $U(\mathbf{a},t)$ is an implicit function of the Eulerian specification. Distances between different flow fields can be computed by Lagrangian specifications.

The Lagrangian specification is easier when using the *flow map* $\Phi : \mathbf{x} \mapsto \Phi_{t_0}^{t_0+t}(\mathbf{x})$, which provides the coordinates of a particle at time t that was at point $\mathbf{x}$ at time t_0. Thus, for given spatiotemperal position, the above mentioned variables can be conveniently expressed as:

$$\Phi_{t_0}^{t_0+t}(\mathbf{x}), p(\Phi_{t_0}^{t_0+t}(\mathbf{x})), T(\Phi_{t_0}^{t_0+t}(\mathbf{x})). \tag{4}$$

3.2 Lagrangian-based Distance Metric

The essence of *e*FLAA is to measure the differences between multiple flow fields in using Lagrangian specification. For the attributes $\mathbf{U}$ and $\mathbf{U}'$ from any two flow fields, without the loss of generality, Lagrangian-based distance at $(\mathbf{x}, t_0)$ in a time span t is defined as:

$$d^t_{\mathbf{x},t_0}(\mathbf{U},\mathbf{U}') = \mu^t_{t_0}(\mathbf{U}(\Phi^{t_0+\tau}_{t_0}(\mathbf{x})), \mathbf{U}'(\Phi^{t_0+\tau}_{t_0}(\mathbf{x}))), \quad (5)$$

where the metric $\mu^t_{t_0}$ measures the distance of the functions $\mathbf{U}$ and $\mathbf{U}'$ over the flow map Φ through time t_0 to t_0+t. The time span t, which is less than the total simulation time T, depends on analysis requirements and potentially also the computing resources. The metric can be flexibly defined, e.g. using the the maximum distance or the Hausdorff distance. $\mathbf{U}$ is usually the identity function for location comparison, or a scalar/vector variable in the data. Geometric quantifications [16], or predicates [36] may also used for different analysis purposes. In the applications of this work, we use the accumulated difference of $\mathbf{U}$ as the metric:

$$d^t_{\mathbf{x},t_0}(\mathbf{U},\mathbf{U}') = \int_{t_0}^{t_0+t} ||\mathbf{U}(\Phi^{t_0+\tau}_{t_0}(\mathbf{x})) - \mathbf{U}'(\Phi^{t_0+\tau}_{t_0}(\mathbf{x}))||^2 d\tau. \quad (6)$$

where $\mathbf{U}$ can be the displacement vector $\mathbf{X}(\mathbf{a},t)$, or scalar quantities defined in Lagrangian specification. In discrete form, the distance (Eq. 6) is computed as:

$$d^t_{\mathbf{x},t_0}(\mathbf{U},\mathbf{U}') = \frac{1}{n}\sum_{k=0}^{n-1} ||\mathbf{U}(\Phi^{t_0+k\Delta t}_{t_0}(\mathbf{x})) - \mathbf{U}'(\Phi^{t_0+k\Delta t}_{t_0}(\mathbf{x}))||^2, \quad (7)$$

where Δt is the size of a time step, and n is the number of time steps.

The distance metric (Eq. 5) brings flexibility for different analysis purposes. For example, we can use the difference of CO_2 concentration along the field line as the metric. Several metrics can be combined to simultaneously take consideration of multiple factors. To generalize, we can also use other field lines, e.g. streaklines to compute distances. In ensemble simulations, our metric not only accounts for the very local domain on the corresponding points, but also counts the nearby samples on the line advection direction.

3.3 Variation Field and Line Filtering

The variation field $\mathscr{V}$ is defined as the average of difference fields:

$$\mathscr{V}(\mathbf{x},t_0,t) = \frac{1}{N(N-1)}\sum_{i<j} d^t_{\mathbf{x},t_0}(\mathbf{U}_i,\mathbf{U}_j), \quad (8)$$

where N is the number of runs, and $d^t_{\mathbf{x},t_0}(\mathbf{U}_i,\mathbf{U}_j)$ is the distance field for run i and j at point $(\mathbf{x},t_0)$. In practice, we use the discrete form of the variation field, which is stored as a time-varying volume. In this volume, variation values are evaluated by interpolation on non-grid points. In general, the resolution of the variation field depends on the granularity of the analysis and the available computation power. The variation field can reveal the ensemble uncertainty of the simulation. As the metric is sensitive to the variations between the ensemble members, our method is capable of capturing even small variations between the flow fields.

There are two ways to visualize the variations of the ensemble flow data, including the direct volume rendering of the variation field and the field line rendering. Although not specific for ensemble run visualization, the field has developed various ways to enable effective visual exploration of complex sets of field lines (e.g. streamlines, pathlines, streaklines). One way is to extract and render only evenly-spaced lines from the vector field, such that the visualization is less cluttered and has less occlusion [20, 42]. Verma and Pang [40] proposed a series of comparative flow visualization methods. Our research does not focus on scalable rendering methods. We leveraged the leading practices, while we dedicated effort on creating more ways to provide overview of the variation field and control of the exploration processes.

In the field line rendering, only a subset of field lines which are capable of depicting differences from flow fields are visualized, in order to avoid visual cluttering. We first find out all the locations where the variation value is greater than a threshold. Then we truncate the corresponding field lines, by dropping the points along the field lines where the variation value is lower than the other thresholds. The line filtering occurs during post-processing process. Although the process is straightforward, the filtered field lines seeded from such locations are likely to represent the top differences between the runs.

3.4 Timeline View

In addition to the variation weighted field lines depicting the spatial distribution of the differences among runs, our system has a timeline view which gives the insights on the simulation differences in an aggregated way. As shown in Fig. 1, there are two main views in the GUI, the 3D navigation and the timeline view. In both views, each color represents one simulation run. The main component in the timeline view is the *difference plot*. The horizontal axis of the difference plot is the time, and the vertical axis encodes the relative distance among runs on each timestep. The timeline view can visualize the regional difference among runs based on the user selection. By default, the timeline view presents the overall differences between ensemble runs. Users can select a region of interest to create a new timeline for a local region. Thus, users can discriminate the quality or the bias of each run in specific locations, in order to study the patterns and improve the simulation models in practice. In the simulation design, usually certain parameters or models are optimized for certain specific geographical region. Our design enables the user to examine the local differences between runs based on their domain knowledge of the models.

The difference plot in the timeline view is generated by 1D Multi-Dimensional Scaling (MDS) projection for all runs. Since the number of runs is relatively low, we use classical MDS [38] to compute the layout in 1D. The distance between each run is the summation of all distances between corresponding points. We denote the distance over the time between two runs as:

$$\mathscr{D}_{i,j}(t) = \int_D d^t_{\mathbf{x},t_0}(\mathbf{U}_i,\mathbf{U}_j)d\mathbf{x}, \quad (9)$$

where D is the spatial domain, and i and j are the indices of runs. A MDS projection is computed to minimize the following function:

$$\min \sum_{i<j} (||x_i(t) - x_j(t)|| - \mathscr{D}_{i,j}(t))^2, \quad (10)$$

where x is the coordinate in the difference plot. The minimum of this equation can be achieved by solving an equivalent eigenvalue problem. The 1D projection result visually separates the runs with higher distances in the horizontal axis, and the trends are also visualized over time. In the timeline view, users can also remove runs considered as "outliers" It is noteworthy that the features that are shown in the spatiotemporal view and the timeline view are complementary but not identical. The timeline view provides a statistical summary of variations as the time evolves, while the spatiotemporal view visualizes a few samples with higher variation values, due to the clutter problems in 3D field line visualization.

4 Scalable Analysis for Ensemble Flows

The computation of the variation field is very expensive due to the line advection and comparison process, which requires a highly scalable system. Unlike previous studies of parallel field line advection, the implication of scalability for this work is unique due to the challenge of data management and memory limits. Independent of ensemble data, the intermediate field line data can be overwhelming — usually 1,000 times larger than the ensemble data at hand. Based on this distinctive scalability concern, we use a redesigned *DStep* framework to maximize the performance given the memory limits.

Scalable field line advection and analysis requires great scalability in both data parallel and task parallel steps. *DStep* provides an elegant glue between the two. Their method is to connect data parallel modules with task parallel modules by using an explicit `<key, value>` construct. This allows optimal scheduling methods to be used for data parallel and task parallel steps orthogonally, without requiring a complicated parallel program that is hard to debug, profile and maintain.

However, our application also has new challenges because the memory limit is the bottleneck of the system. Let us now overview the new needs and then briefly introduce our architecture.

4.1 System Design

In comparison to visualizing ensemble scalar fields, ensemble flow fields have received far less attention. This is partly due to the expense of computing, storing, comparing and visualizing unsteady flow features in the form of fieldlines.

Scalable Performance: Existing works typically use Euler specification for distance metrics, instead of the computationally more expensive but more proper Lagrangian specification. However, our work of *e*FLAA needs to use Lagrangian specification for full range analysis of the ensemble flow variation.

The *e*FLAA system needs to implement synchronized field line extraction across all ensemble runs simultaneously, which has not been attempted in previous works. Synchronized field line extraction removes the need to store all field lines to disk, thereby omitting an otherwise unaffordable cost. This will enable comprehensive analysis of the variations in unsteady flows "in-situ". The challenge lies in better load-balance control with an even more unpredictable workload distribution. The result of synchronized field line extraction is a high-resolution and reusable spatiotemporal variation field that then guides subsequent interactive visualization at much lower computational cost.

Scalable Data Management: Current visualization systems usually compute and store the field lines to disk, in a run by run fashion through the entire ensemble. The field lines are then read back into memory for analysis. The overwhelming I/O costs, however, will greatly limit the scale of the problem that can be practically studied. As documented by [17], I/O cost for parallel visualization could take up to 90% of the entire computing time.

A more serious challenge is the memory footprint in the synchronized field line extraction of all runs. The field lines for computing the variation field requires at least tracing one field line from each spatiotemporal location in every ensemble run. In the GEOS-5 data for example, every vertex includes seven floating point values for 4D spatiotemporal position and 3D velocity. The models are on a $288 \times 181 \times 72$ spatial grid with 24 monthly time steps and 8 runs in the ensemble. The total size of the loaded data is around 13GB. A field line is terminated after one of the following three criteria is met: (1) more than 1000 vertices (4th order adaptive Runge-Kutta), (2) the field line has lasted more than one month of wall-clock time, or (3) a critical point has been reached. At a maximum (not considering criteria 2 and 3), the field lines will together take up to 17.5 TB. When also considering criteria 2 and 3, we learned from experience that field lines are on average three times shorter. In result, a practical estimation is that the field lines for the analysis will take up to 5.86TB to store. From 13GB to 5.86TB is an increase of $3,000\times$. That literally means, without our data management, one would need to have a petascale supercomputer just to analyze a terascale ensemble simulation, which is an unaffordable cost for ensemble flow analysis.

4.2 Basics of DStep Framework

Now let us briefly review key concepts in *DStep* framework. Similar to MapReduce routines, without explicit management of complex communication and job scheduling, application developers only need to implement `dstep()` (`map()`) and `reduce()` functions with proper key-value pairs. Domain traversal is simplified by recursively calling `emit_dstep()` functions, which queues a job to continue the unfinished work. In field line tracing for example, each process is in charge of a local domain. Partial line tracing is implemented in `dstep()`. When the trace goes out of the local domain, a new line tracing job is emitted to continue the tracing. After the trace is finished, the partial lines are merged in `reduce()` function.

Architecture wise, *DStep* uses a two-tiered job management. The first tier is by data partition and hence manages the data parallel aspect. The second tier handles the task parallel aspect. It is based on multiple task queues, i.e independent queues dedicated to different types of tasks. On each processing node in a supercomputer, there are typically a number of processor cores. On the cores within a node, *DStep* will set up a group of worker threads. These threads play different roles, such as `steppers`, `reducers`, `writers`, and `communicators`. Intra-group communication is local to a node, while inter-group messages are routed through and aggregated by communicators for efficiency. More details on *DStep* framework is available in previous literature [18]. We redesign the framework in order to fit our goals.

4.3 Parallel Computing of the Variation Field

In comparison to *DStep*, *e*FLAA adds significant new functionality in the job management module to simultaneously control memory footprint and improve load balance. This is because field lines together can take $1,000$ times more space than the unsteady flow field itself. Without dynamic control of runtime memory footprint, scalable uncertainty analysis for an entire ensemble is unfeasible. Details of this addition are presented in Section 4.4.

The computing of variation fields is quite straightforward, as illustrated in Fig. 4. The system first loads all data blocks for all the runs from the parallel file system. The spatiotemporal domain is decomposed and distributed in the same way used by *DStep*.

Field lines are extracted by `steppers`. The key-value pair is `<point, partial_line>`. Seeds are inputed as the key `point` from the batch manager for field line initialization. Adaptive Runge-Kutta 4th order numerical integration is used to trace the lines. If an incoming point is out of the local boundary of data for a worker, a new intermediate step job is emitted to the system to continue the work later by other workers. Simultaneously, the partial field line is send to the reducer for the further processing.

Variation values as defined in Section 3 are computed by `reducers`. Each reducer handles partial fields in sets, where a set consists of partial fieldlines originated from the same seed location, regardless which run they are from. The field lines are merged, and then re-sampled to consistent time intervals for distance and variation computation. On the exit of the pipeline, the variation field is also stored into the file system, which are used for line truncation and visualization. The pseudo code of the `reduce()` function is as follows:

function REDUCE(seed, partial_fieldlines[])
  sort partial_fieldlines[] into partial_fieldlines$_i$[] by run_id
  for $i = 1 \rightarrow N$ **do**
    sort partial_fieldlines$_i$[] by hops
    fieldline$_i$ = merge_fieldline(partial_fieldlines$_i$[])
  end for
  for $i = 1 \rightarrow N$ **do**
    for $j = 1 \rightarrow N$ **do** ▷ *Compare fieldlines from every run*
      $d_{i,j}$(seed) = distance(fieldline$_i$, fieldline$_j$)
    end for
  end for
  $\mathscr{U}(\text{seed}) = \frac{1}{N(N-1)} \sum_{i<j} d_{i,j}(\text{seed})$
  if $\mathscr{U}(\text{seed}) >$ threshold **then** ▷ *Only save the fieldlines that are with high variation*
    for $i = 1 \rightarrow N$ **do**
      emit_write(seed, fieldline$_i$)
    end for
  end if
end function

In the end, the output is a relatively small set of representative field lines together with a high resolution temporal volume of variations. The massive intermediate data, which are much larger than the input data, are processed and discarded on-the-fly.

4.4 Batch Streaming

The field line data is often too large (even larger than the raw flow field data) to store and subsequently load it. Our approach provides an "in-situ" way to compare and filter field lines in the reduce stage.

The management for in-situ analysis uses batch streaming. We couple field line extraction by `steppers` and field line reduction and variation computation by `reducers` in to "cycles". In each cycle, only a batch of seeds are pushed into the pipeline. For each batch,

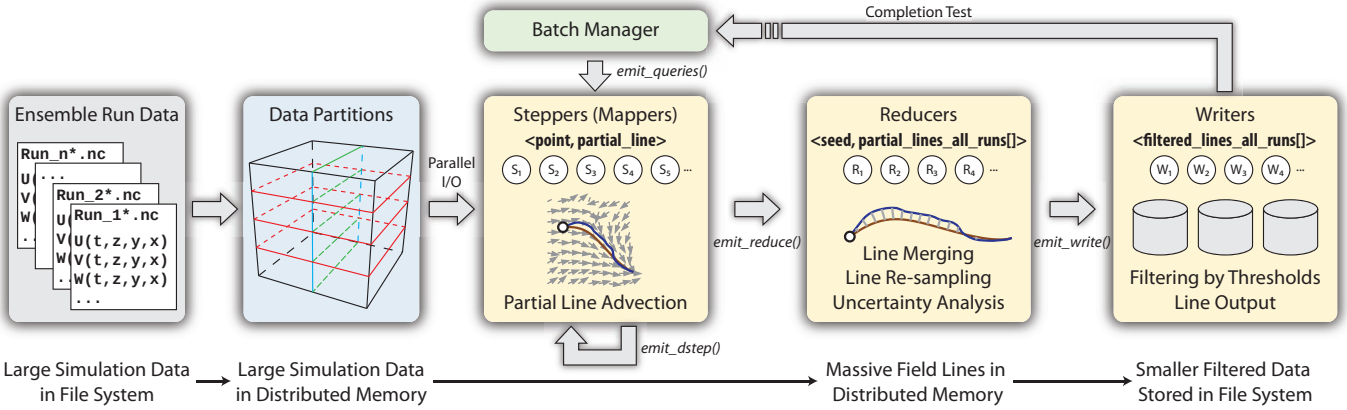

Fig. 4. The parallel pipeline of proposed system, based on the Map-Reduce like DStep framework. The raw ensemble run data is partitioned and loaded into the system at the beginning, and then the computation is conducted in `step` and `reduce` stages in parallel. In the `step` stage, the partial field lines are traced, and they are later merged, re-sampled, and analyzed in the `reduce` stage, before filtered lines are written. A batch manager is used to control the workload and the memory footprint of the pipeline.

we ensure all intermediate results with the same key (seed) are reduced together, such that the variation for the spatiotemporal location in that ensemble is computed. This policy may slightly slow down the pipeline, but the memory footprint can be controlled, given a relatively large number of seeds. Both the `steppers` and `reducers` are computationally intensive. Steppers handle field line advection. Reducers are multi-threaded and handle merging and re-sampling of field lines, computing the distances between all runs. Through experimenting, we found it optimal for each worker group to have equal number of steppers and reducers.

Batch size is defined in number of seeds, i.e. number of field lines in the batch. The smallest is to have one field line per batch. This will lead to unacceptable performance scalability, in part because *e*FLAA uses static data partition and distribution on the node-level. The largest is to have all the field lines in the same batch. This will lead to unacceptable space scalability as noted in the above. The balance between performance and space scalability is crucial to *e*FLAA, and the point of balance is machine dependent. For the two test machines, we empirically chose the batch size to be 80k and 10k field lines, respectively.

4.5 Reusable Ensemble Flow Variation Field

*e*FLAA outputs the variation field and only a selected subset of representative field lines. Since users only care about the field lines that can characterize the differences between runs, only a part of the field lines are selected (using thresholding) and stored in the file system. Our system also allows users to reuse the computed variation field to generate more field lines on-the-fly. The existing variation fields can help with the line filtering without rerunning the whole pipeline. In addition, field line tracing is greatly accelerated because the problem size is much reduced. This reusable ensemble variation field is useful in a few different situations. For scientists who only have limited computing resources, re-using the variation fields is a good choice to achieve the goal with less cost. For post-analysis purposes, the variation field can be computed with supercomputers, and further analysis can be conducted on smaller workstations or even desktop computers.

The re-use of the variation fields is also helpful for reduceing I/O cost during subsequent analysis. Only limited numbers of field lines are required to be traced again. Thus, the patterns of grid cell access are limited and predictable. We implemented an experiment to validate this hypothesis (Fig. 5). In the experiment, we re-traced field lines from seed locations with top 2k, 4k and 8k variation values. The result demonstrates that only a small portion of grid cells that are visited. Subsequent analysis should indeed be able to implement sparse-volume representation, as opposed to storing the full 4D array of voxels. This will greatly reduce needs of computing, memory and I/O bandwidth, thereby allowing machines that are far smaller than a supercomputer to practically handle full-scale and full-range analysis. Our technique thus greatly increases the potential for widespread penetration of ensemble flow analysis in disparate application domains.

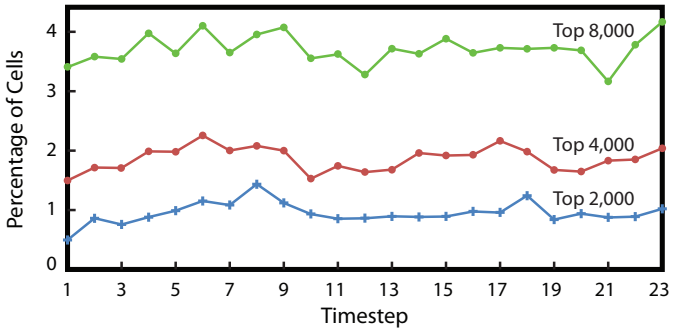

Fig. 5. The cell access patterns of re-using variation field by tracing field lines from seeds with top variation values in GEOS-5 simulation data. The horizontal axis is the time step, and the vertical axis is the percentage of grid cells that are accessed.

4.6 Parallel Scalability

We evaluate the system performance with several datasets on two supercomputers in National Super Computer Center in Jinan (NSCCJN), which is located in Shandong province, China. The hardware architecture of the two supercomputers are ShenWei and x86, respectively. The microprocessors of the former one are SW1600, which are developed and produced in China. The performance is also evaluated in the x86-based supercomputer in the same center.

The ShenWei-based supercomputer consists of 8,704 16-core SW1600 processors, which operate at 1.0-1.1Ghz. The theoretical peak floating point performance of each SW1600 processor can achieve 128Gflops. Each physical node in the racks is equipped with 1 processor and 16GB memory, and is further divided into 4 virtual nodes equally. Thus, each virtual node is allocated with 4 ShenWei cores and approximately 3GB memory. The bandwidth of the high-speed network is 40Gbps. The operating system is 64-bit Linux, and the parallel file system is high-performance SWGFS, which is developed in China. Specialized MPI and C/C++ compilers are provided for the ShenWei architecture.

The x86-based supercomputer in NSCCJN is composed of 700 nodes, each of which has 2 hexa-core Intel Xeon E5675 processors working at 3.06GHz. Our allocation can use 10% of the computing resources. The main memory for each node is 36GB. The nodes are connected with InfiniBand QDR interfaces, whose theoretical bandwidth is 40Gbps. The parallel file system of this supercomputer share the same SWGFS with the ShenWei-based machine. MVAPICH2 compilers are provided for interprocess communication.

Customized software configuration and job layouts are designed ac-

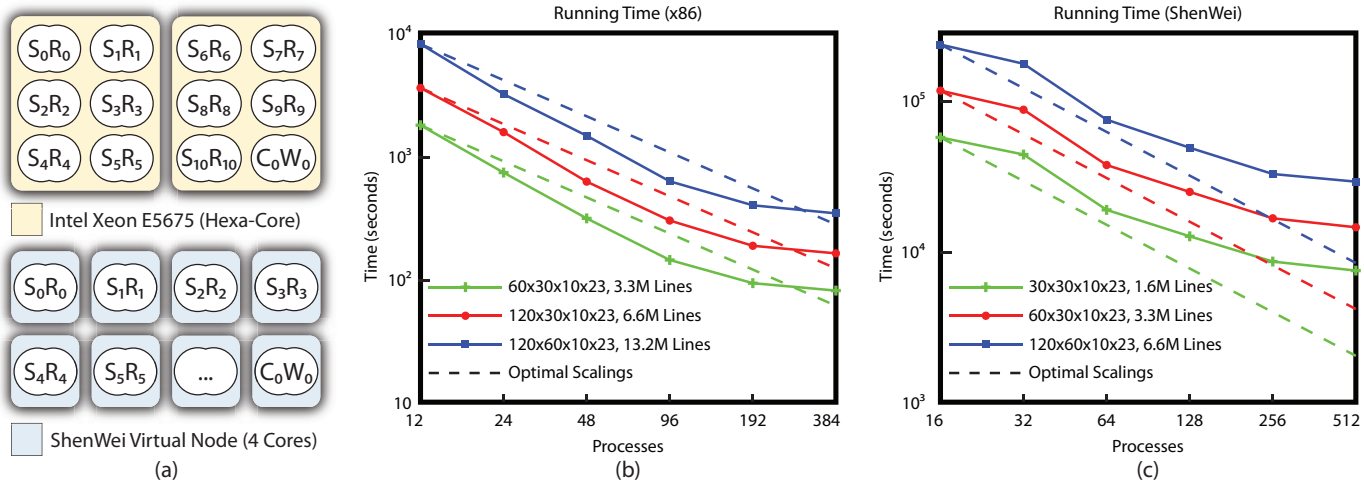

Fig. 6. The worker group configuration and the timings on both x86 and ShenWei architectures. Each worker group contains a communicator-writer process and several stepper-reducer processes. On x86 architecture, each process occupies two hexa-core CPUs on a node; on ShenWei, each process takes a virtual node with 4 cores. Performance of several problem sizes are tested on both x86- and ShenWei-based supercomputers.

cording to the hardware architectures and our application. The DStep job layout is shown in Fig. 6(a). For ShenWei supercomputer, because there is less than 4GB memory for each virtual node (the operating system occupies some space), it is insufficient to run 4 DStep processes on one single virtual node. Instead, we run only 1 process per virtual node, and 3 threads to accelerate stepper and reducer jobs. In each worker group, there are 15 stepper-reducer processes and 1 communicator-writer process. For the x86-based supercomputer, we use 11 stepper-reducer processes and 1 communicator-writer process.

The timings of the system with different numbers of processes on both architectures are shown in Fig. 6. An 8-run GEOS-5 dataset is used to evaluate the performance and the scalability. The grid size of the data is $288 \times 181 \times 72$, with 24 time steps. The monthly-averaged multivariate ensemble-run dataset is stored in netCDF format across 192 files with float-precision, which totals 76GB. 5 variables which are essential to compute the velocities and distances are loaded, so the effective data size to load is about 13.4GB. The whole spatiotemporal domain is partitioned into 512 data blocks for computation, which are assigned to all steppers in round-robin manner.

In our experiment, different problem sizes are conducted by specifying various variation field resolutions, which also determines the number of field lines that are traced. The timing results shown in Fig. 6 present the scalability of the system on both platforms. On x86-based machines, as the number of processes increases, super-linear speedup is observed from 12 (1 worker group) processes to 192 processes (16 worker groups). The super-linear speedup is likely caused by the communication congestion and the high memory occupation. On ShenWei-based supercomputer, we tested timings on up to 512 processes, which uses 2048 cores. Good scalability is observed in the timings. Both platforms show linear speed-up as the data scale increases for all numbers of processes.

The system is also portable to supercomputers with different architectures. The batch size needs to be modified according to the memory limit of the nodes. Generally, the larger the memory, the higher the throughput and performance. Other parameters, (e.g. the worker group size, the workload for epochs) also need to be tuned according to the network bandwidth and performance of the processors.

5 Application Results

5.1 GEOS-5 Simulation Data

GEOS-5 is a state-of-the-art atmospheric model from NASA Goddard Space Flight Center. It aims at better understanding the internal-model and applying teleconnection analysis. A flux-form semi-Lagrangian finite-volume dynamical core with floating vertical coordinates [19] is used in the GEOS-5 general climate model (GCM). A variety of attributes, including divergence, vorticity, wind speed and pressure, are computed with GCM. The output of the simulation is stored in hybrid-sigma pressure grid, with the spatial resolution of $1^\circ \times 1.25^\circ$ with 72 pressure levels in the vertical direction, which ranges from 1 atm (near to the terrain surface) to 0.01 hPa (about 80 km).

Previously, the scientists examined the internal model variability of GEOS-5 by running an 8-run ensemble simulation [2, 18], and found large variations between the ensemble members due to high sensitivity to the transportation model. In our experiment, we further examine the variations of the transportation as well as the CO_2 concentration of fossil fuel emissions on the wind paths with Lagrangian-based metrics. We follow the previous research to use the monthly average data of the 8 runs from Janurary, 2000 to December, 2011 [2]. There are 24 time steps saved in separate netCDF files, containing 35 variables in floating-point precision. The data for all 8 runs amounts to about 76GB. Our Runge-Kutta advection has been customized for the curvilinear and hybrid-sigma pressure grid.

The visualization results with our method are presented in Fig. 1. Two time steps are to show the differences of the ensemble runs. For example, in March, 2000, the traces originating from the East Pacific Ocean near the equator are much different. While in most runs, the mainstream trends of the traces are toward to the west along the equator, some of the runs are intriguing. In Run #4, the trace went to the west-south and then further proceed to the Australian continent. In contrast, Run #8 is more intriguing. It first went toward the west, and then made a U-turn to go toward the Carribean Sea. The dramatic changes of directions in this month can also be discovered in this view, e.g. the North America and the south Pacific Ocean, etc. In March, 2001, some interesting swirl patterns can be observed in North America in some runs, which are significantly different to other runs.

We also compared the results with two different Lagrangian-based distance metrics in Fig. 7, in order to examine the sensitivity of CO_2 concentration due to the transportation [2]. The first metric only accounts for the location differences, and the second one only considers the differences of CO_2 fossil fuel concentration. Intermediate variation fields are also visualized with volume rendering for illustration. We follow Equation 5 to compute the Lagrangian-based distance and uncertainty of CO_2 fossil fuel concentration along the pathlines. In Fig. 7, 3 consecutive months (From January to March, 2000) are shown to compare the two metrics. In January, although the results from the location-based metric appears to diverge from run to run, the CO_2 fossil fuel concentration along the traces is not varied in certain regions, like the U.S. region. Along the equator and several locations in the Southern Hemisphere, higher variation values are observed. In earth system, the distribution of CO_2 mainly depends on two factors, including the wind transportation and the emission/absorption on the land. In this simulation, CO_2 fossil fuel concentration is relatively

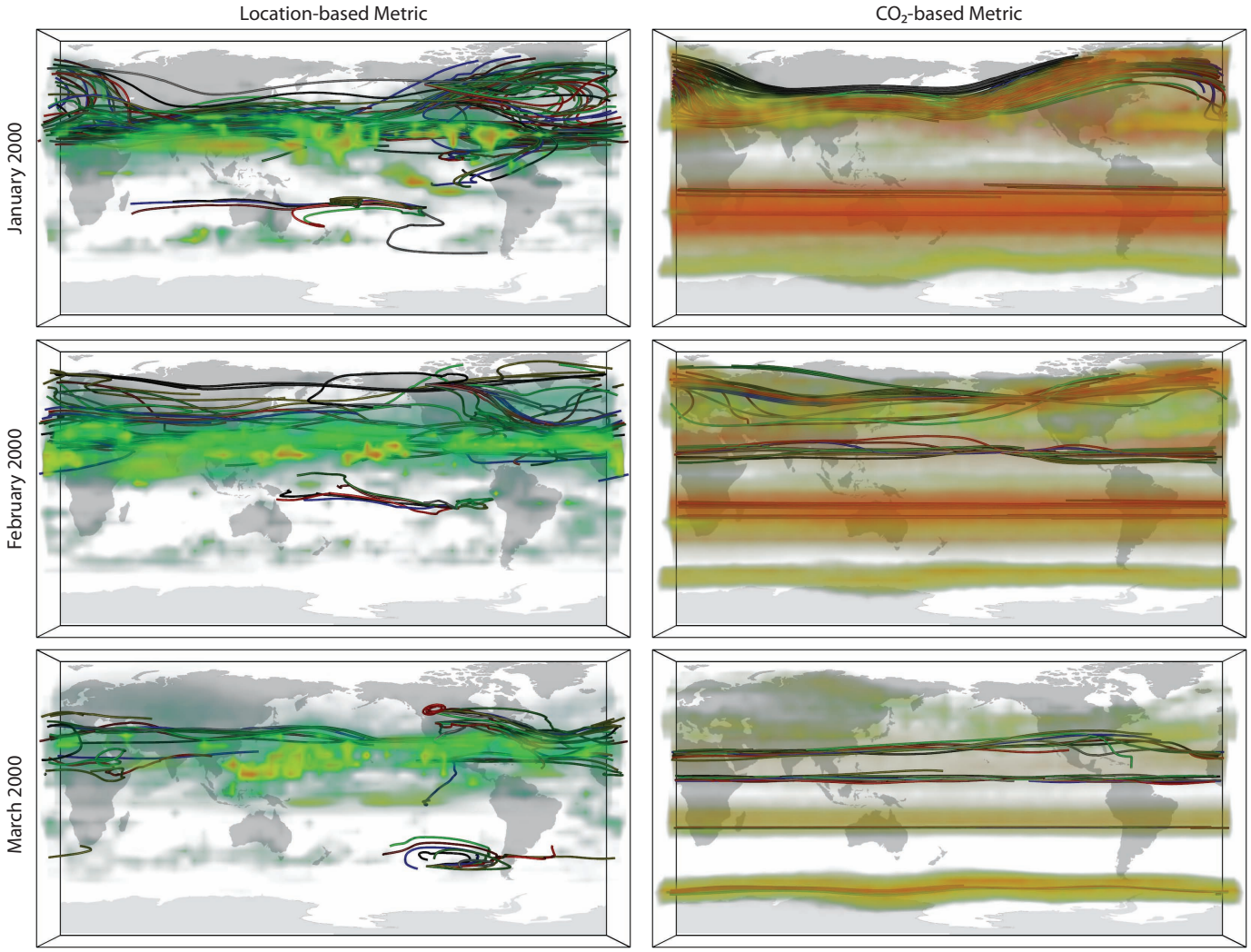

Fig. 7. The volume rendered variation field and the filtered field lines of three consecutive months of GEOS-5 simulation data. On the left column, location-based metric is used to compute the variation field while CO2-based metric is used in the right column. Features of interests are visualized in the results to reveal the differences between the ensemble runs.

stable, so it appears to be insensitive to the initial values. The observations from our results further validate the conclusions in previous research. With flexible user-defined distance metrics, useful traces can be filtered to show the differences of the ensemble runs. Our system allows scientists to flexibly investigate variations of their models.

5.2 WRF Simulation Data

In this application, the researchers in environmental science would like to investigate how urbanization influences the weather and climate. Urbanization is usually considered to be influential to the change of climates, yet the complicated mechanisms are still open research problems. Ensemble simulation with different initialization data is one way to study this important topic.

In the experiment, the scientists conducted two runs with the WRF (Weather Research and Forecasting) model, which is a well known and routinely used numerical prediction system for weather research and forecasting [1]. The first run, namely the base run, is initialized with actual data. In the other run, the urban area is replaced by vegetation landuse. The scientists who conducted the simulation would like to see how weather differs with the two distinct geographical conditions.

The simulation starts from 2012-7-1 00:00:00 UTC to 2012-7-10 18:00:00 UTC, and spatially covers East China. The dimension of the grid is 100 (west-to-east) $\times$ 100 (south-to-north) $\times$ 27 (vertical pressure layers). Hourly data is stored in the simulation, and the overall output data is about 4GB for each run. In our analysis, the three wind components, as well as the essential variables to convert coordinates in vertical layers are used. Other variables can be loaded depending on the analysis purposes.

The visualization results of WRF data are shown in Fig. 8. The 3D rendering includes the volume rendered variation field and the filtered pathlines. In the time line view, the overall differences between the two runs are shown. At the very beginning, the two runs are very close to each other. There are only small differences in wind directions in the north. 40 hours later, there is a peak in the overall difference. The traces originating from the north lead to considerable displacement in the coastline of China. But later, it is interesting to see the distances decrease. In the late stage of the simulation, another peak appears. The variation values of the majority of the data domain increase. Moreover, in the entire sequence, the sea region presents lower variation than the mainland since sea surface temperatures are identical for both simulations. It is also worth noticing that during the second peak, the regions with higher variation are moving toward the east, as shown in the bottom of Fig. 8. Although the impact of urbanization on the weather seems not to be significant in this short period of time, our tool is still capable of detecting small differences and captures the overall trend of 3-dimentional influence patterns in this case.

Positive feedback is given by the scientists who conducted this simulation. The visualization is very intuitive and straightforward to them. The timeline view is very useful for the researchers to compare and evaluate the ensemble runs. It is intriguing to see the two peaks in the overall trends. The extracted lines and the variation field are quite

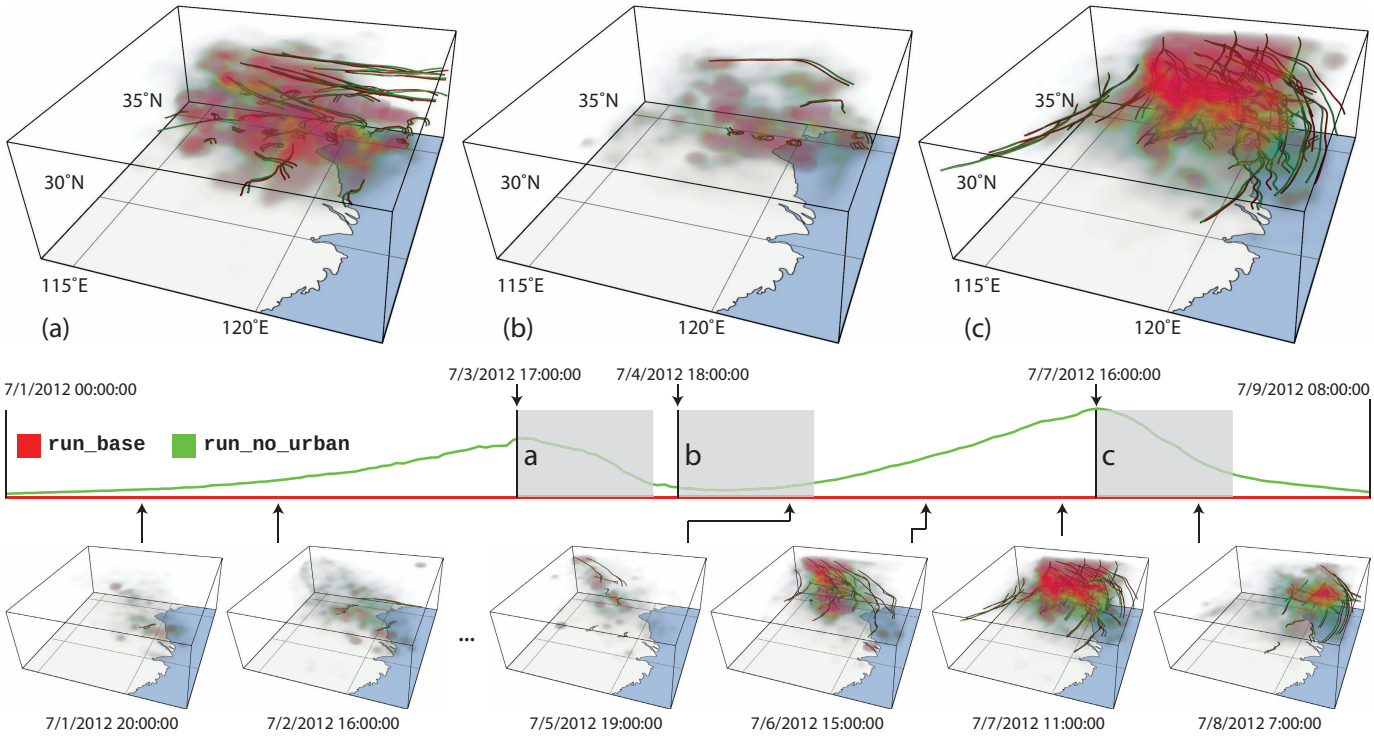

Fig. 8. The volume rendered variation field and the filtered field lines of the WRF simulation data. The difference curves over the time (UTC) are shown in the timeline view.

clear to them and useful to investigate the influences of urbanization. Without our tool, it is hard for them to compare the differences between runs effectively. Although in the current application, it is far from drawing conclusions of the impact as to urbanization on weather, more ensemble simulations will be conducted and visualized with our tool to further investigate this complex research topic.

6 User Feedback and Discussions

We have received positive feedback from domain experts in climate and environmental sciences. The scientist who conducted the WRF simulations finds the differences shown in the visualization clear and useful. There is a continued lack of analysis tools that can automatically extract and reveal field line differences in ensemble simulation runs. The emergence, migration, and dissipation of variation patterns can help to design and refine future study of the influences of urbanization. Another scientist who specializes in climate modeling and model evaluation often runs tens of ensemble runs to study the models. The most important measurements of model difference are bias and diversity. The scientist feels our tool is useful for revealing model diversity in unsteady flow components.

Both scientists like the timeline view. The peaks shown in the WRF simulation results clearly show the differences between runs. In addition, intuitively visible are large scale variation patterns that seem to be periodic. Climate modelers pay attention to model effectiveness and model validation on selected geographic regions. Our design of region-selectable timeline view directly serves that need.

The Lagrangian-based variation is more effective than the traditional routinely-used local statistical-based uncertainty metrics, which are primarily Eulerian-based. This is exemplified by the case of synthetic 2D unsteady flow, with Gaussian noise injected into the flow field. While Lagrangian-specification elegantly handles the analysis task, using Eulerian-based metrics leads to an almost unusable uncertainty analysis. This reference case clearly demonstrates the importance of Lagrangian-based variation.

Our variation metric can be generally adapted to different applications. For the Lagrangian-based distance metric, users can choose to use difference of position, scalar values or other combinations. In GEOS-5 application, we demonstrated how CO_2 concentration is used to compute the pairwise distance between the samples on the field lines, and then obtain the variation value by summation. Such user-defined application-driven metrics can be introduced directly from the domain science to solve research problems in the domain science.

7 Conclusions and Future Work

Although our work on *e*FLAA is primarily concerned with analysis methods, it requires a parallel solution that is scalable in performance and, more importantly, in space. As described in Section 4.4, the use of a petascale supercomputer to analyze a terascale ensemble simulation is prohibitive for the vast majority of scientists. *e*FLAA solves this problem by closely coupling field line extraction with field line analysis, and using batch streaming to control the entire process. Furthermore, after the variation field has been computed, users' analysis can be done very flexibly and quickly. For example, when hypothesizing about model diversity specific to a geographic region, one can easily create test cases by filtering for that region and also filtering for high flow variation. Due to this, our variation field is "reusable".

As future work, we plan to extend our system to support irregular grids. *e*FLAA currently handles rectilinear and hybrid-sigma grid (e.g. GEOS-5 models). The current way of seeding pathlines is by uniform sampling in the domain, which is of course not the optimal way for our goal. In the future, we will leverage multi-resolution techniques to produce more effective and efficient analysis. Interaction techniques are also necessary for the coupled analysis on ensemble flow data. The field also needs new methods for analyzing the variation of ensemble tensor fields in future.

Acknowledgments

The authors wish to thank anonymous reviewers for their comments. This work is supported by NSFC No. 61170204. This work is also partially supported by NSFC Key Project No. 61232012 and the "Strategic Priority Research Program - Climate Change: Carbon Budget and Relevant Issues" of the Chinese Academy of Sciences Grant No. XDA05040205. The authors would like to thank Dr. Junfeng Liu for providing WRF simulation data and feedback, and thank Dr. Xiaoguang Ma for his suggestive comments.

References

[1] The WRF model. http://wrf-model.org.

[2] M. Allen, D. Erickson, W. Kendall, J. Fu, L. Ott, and S. Pawson. The influence of internal model variability in GEOS-5 on interhemispheric CO2 exchange. *Journal of Geophysical Research: Atmospheres*, 117(D10):107:1–20, 2012.

[3] W. Berger, H. Piringer, P. Filzmoser, and E. Gröller. Uncertainty-aware exploration of continuous parameter spaces using multivariate prediction. *Comput. Graph. Forum*, 30(3):911–920, 2011.

[4] R. P. Botchen and D. Weiskopf. Texture-based visualization of uncertainty in flow fields. In *Proceedings of IEEE Visualization 2005*, pages 647–654, 2005.

[5] S. Bruckner and T. Möller. Result-driven exploration of simulation parameter spaces for visual effects design. *IEEE Trans. Vis. Comput. Graph.*, 16(6):1468–1476, 2010.

[6] D. Camp, C. Garth, H. Childs, D. Pugmire, and K. I. Joy. Streamline integration using MPI-hybrid parallelism on a large multicore architecture. *IEEE Trans. Vis. Comput. Graph.*, 17(11):1702–1713, 2011.

[7] C.-M. Chen, L. Xu, T.-Y. Lee, and H.-W. Shen. A flow-guided file layout for out-of-core streamline computation. In *Proceedings of IEEE Pacific Visualization Symposium 2012*, pages 145–152, 2012.

[8] L. Chen and I. Fujishiro. Optimizing parallel performance of streamline visualization for large distributed flow datasets. In *Proceedings of Pacific Visualization Symposium 2008*, pages 87–94, 2008.

[9] G. Evensen. Sequential data assimilation with nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research*, 99(C5):143–162, 1994.

[10] C. Garth, F. Gerhardt, X. Tricoche, and H. Hagen. Efficient computation and visualization of coherent structures in fluid flow applications. *IEEE Trans. Vis. Comput. Graph.*, 13(6):1464–1471, 2007.

[11] M. Gleicher, D. Albers, R. Walker, I. Jusufi, C. D. Hansen, and J. C. Roberts. Visual comparison for information visualization. *Information Visualization*, 10(4):289–309, 2011.

[12] M. Hlawatsch, P. Leube, W. Nowak, and D. Weiskopf. Flow radar glyphs - static visualization of unsteady flow with uncertainty. *IEEE Trans. Vis. Comput. Graph.*, 17(12):1949–1958, 2011.

[13] C. Johnson. Top scientific visualization research problems. *IEEE Comput. Graph. Appl.*, 24(4):13–17, 2004.

[14] C. R. Johnson and A. R. Sanderson. A next step: Visualizing errors and uncertainty. *IEEE Comput. Graph. Appl.*, 23(5):6–10, 2003.

[15] J. Kehrer, P. Muigg, H. Doleisch, and H. Hauser. Interactive visual analysis of heterogeneous scientific data across an interface. *IEEE Trans. Vis. Comput. Graph.*, 17(7):934–946, 2011.

[16] W. Kendall, J. Huang, and T. Peterka. Geometric quantification of features in large flow fields. *IEEE Comput. Graph. Appl.*, 32(4):46–54, 2012.

[17] W. Kendall, J. Huang, T. Peterka, R. Latham, and R. Ross. Visualization viewpoint: Towards a general I/O layer for parallel visualization applications. *IEEE Comput. Graph. Appl.*, 31(6):6–10, Nov./Dec. 2011.

[18] W. Kendall, J. Wang, M. Allen, T. Peterka, J. Huang, and D. Erickson. Simplified parallel domain traversal. In *SC11: Proceedings of the ACM/IEEE Conference on Supercomputing*, pages 10:1–10:11, 2011.

[19] S.-J. Lin. A "vertically Lagrangian" finite-volume dynamical core for global models. *Monthly Weather Review*, 132(10):2293–2307, 2004.

[20] Z. Liu, R. J. Moorhead, and J. Groner. An advanced evenly-spaced streamline placement algorithm. *IEEE Trans. Vis. Comput. Graph.*, 12(5):965–972, 2006.

[21] S. K. Lodha, A. Pang, R. E. Sheehan, and C. M. Wittenbrink. UFLOW: Visualizing uncertainty in fluid flow. In *Proceedings of IEEE Visualization 1996*, pages 249–254, 1996.

[22] A. L. Love, A. Pang, and D. L. Kao. Visualizing spatial multivalue data. *IEEE Comput. Graph. Appl.*, 25(3):69–79, 2005.

[23] B. Nouanesengsy, T.-Y. Lee, K. Lu, H.-W. Shen, and T. Peterka. Parallel particle advection and FTLE computation for time-varying flow fields. In *SC12: Proceedings of the ACM/IEEE Conference on Supercomputing*, pages 61:1–61:11, 2012.

[24] B. Nouanesengsy, T.-Y. Lee, and H.-W. Shen. Load-balanced parallel streamline generation on large scale vector fields. *IEEE Trans. Vis. Comput. Graph.*, 17(12):1785–1794, 2011.

[25] M. Otto, T. Germer, and H. Theisel. Uncertain topology of 3D vector fields. In *Proceedings of Pacific Visualization 2011*, pages 67–74, 2011.

[26] H.-G. Pagendarm and F. H. Post. Comparative visualization – approaches and examples. In *Proceedings of Visualization in Scientific Computing*, pages 95–108, 1995.

[27] A. Pang, C. M. Wittenbrink, and S. K. Lodha. Approaches to uncertainty visualization. *The Visual Computer*, 13(8):370–390, 1997.

[28] T. Peterka, R. B. Ross, B. Nouanesengsy, T.-Y. Lee, H.-W. Shen, W. Kendall, and J. Huang. A study of parallel particle tracing for steady-state and time-varying flow fields. In *IPDPS11: Proceedings of the International Parallel and Distributed Processing Symposium*, pages 580–591, 2011.

[29] C. Petz, K. Pöthkow, and H.-C. Hege. Probabilistic local features in uncertain vector fields with spatial correlation. *Comput. Graph. Forum*, 31(3):1045–1054, 2012.

[30] T. Pfaffelmoser, M. Reitinger, and R. Westermann. Visualizing the positional and geometrical variability of isosurfaces in uncertain scalar fields. *Comput. Graph. Forum*, 30(3):951–960, 2011.

[31] K. Pöthkow and H.-C. Hege. Positional uncertainty of isocontours: Condition analysis and probabilistic measures. *IEEE Trans. Vis. Comput. Graph.*, 17(10):1393–1406, 2011.

[32] K. Potter, J. Kniss, R. F. Riesenfeld, and C. R. Johnson. Visualizing summary statistics and uncertainty. *Comput. Graph. Forum*, 29(3):823–832, 2010.

[33] K. Potter, P. Rosen, and C. R. Johnson. From quantification to visualization: A taxonomy of uncertainty visualization approaches. In *Uncertainty Quantification in Scientific Computing*, volume 377, pages 226–249. Springer Berlin Heidelberg, 2012.

[34] K. Potter, A. T. Wilson, P.-T. Bremer, D. N. Williams, C. M. Doutriaux, V. Pascucci, and C. R. Johnson. Ensemble-vis: A framework for the statistical visualization of ensemble data. In *ICDM 2009: Proceedings of IEEE International Conference on Data Mining Workshops*, pages 233–240, 2009.

[35] D. Pugmire, H. Childs, C. Garth, S. Ahern, and G. H. Weber. Scalable computation of streamlines on very large datasets. In *SC09: Proceedings of the ACM/IEEE Conference on Supercomputing*, pages 16:1–16:12, 2009.

[36] T. Salzbrunn and G. Scheuermann. Streamline predicates. *IEEE Trans. Vis. Comput. Graph.*, 12(6):1601–1612, 2006.

[37] J. Sanyal, S. Zhang, J. Dyer, A. Mercer, P. Amburn, and R. J. Moorhead. Noodles: A tool for visualization of numerical weather model ensemble uncertainty. *IEEE Trans. Vis. Comput. Graph.*, 16(6):1421–1430, 2010.

[38] W. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4):401–419, 1952.

[39] A. Unger and H. Schumann. Visual support for the understanding of simulation processes. In *Proceedings of IEE Pacific Visualization Symposium 2009*, pages 57–64, 2009.

[40] V. Verma and A. Pang. Comparative flow visualization. *IEEE Trans. Vis. Comput. Graph.*, 10(6):609–624, 2004.

[41] C. M. Wittenbrink, A. Pang, and S. K. Lodha. Glyphs for visualizing uncertainty in vector fields. *IEEE Trans. Vis. Comput. Graph.*, 2(3):266–279, 1996.

[42] K. Wu, Z. Liu, S. Zhang, and R. J. M. II. Topology-aware evenly spaced streamline placement. *IEEE Trans. Vis. Comput. Graph.*, 16(5):791–801, 2010.

[43] H. Yu, C. Wang, and K.-L. Ma. Parallel hierarchical visualization of large time-varying 3d vector fields. In *SC07: Proceedings of the ACM/IEEE Conference on Supercomputing*, pages 24:1–24:12, 2007.