Feature-Based Statistical Analysis of Combustion Simulation Data

Janine C. Bennett, *Member, IEEE*, Vaidyanathan Krishnamoorthy, Shusen Liu, Ray W. Grout, Evatt R. Hawkes, Jacqueline H. Chen, Jason Shepherd, Valerio Pascucci, *Member, IEEE*, and Peer-Timo Bremer, *Member, IEEE*

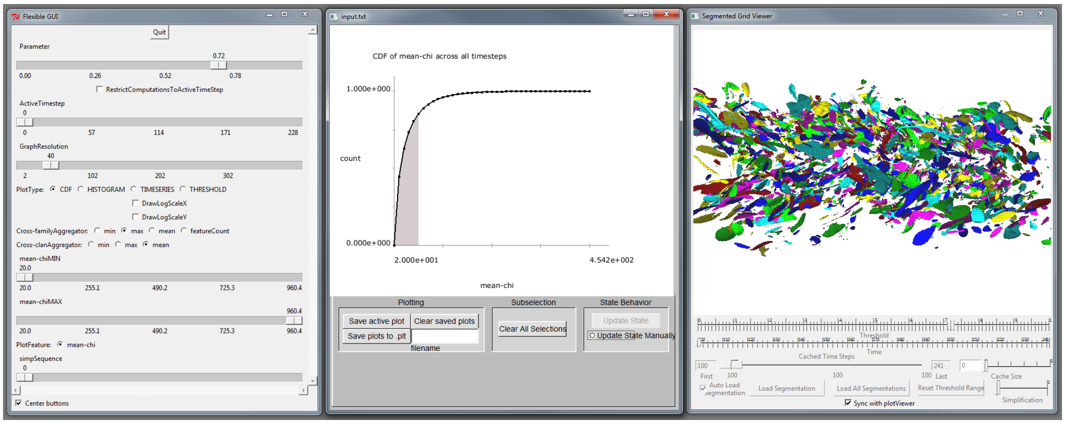


Fig. 1. Our framework provides a natural and intuitive work-flow for the exploration of global trends in feature-based statistics. By efficiently encoding hierarchical meta-data in a pre-processing step, interactive data exploration of the equivalent of one terabyte of simulation data is performed on a commodity desktop.

Abstract—We present a new framework for feature-based statistical analysis of large-scale scientific data and demonstrate its effectiveness by analyzing features from Direct Numerical Simulations (DNS) of turbulent combustion. Turbulent flows are ubiquitous and account for transport and mixing processes in combustion, astrophysics, fusion, and climate modeling among other disciplines. They are also characterized by coherent structure or organized motion, *i.e.* nonlocal entities whose geometrical features can directly impact molecular mixing and reactive processes. While traditional multi-point statistics provide correlative information, they lack nonlocal structural information, and hence, fail to provide mechanistic causality information between organized fluid motion and mixing and reactive processes. Hence, it is of great interest to capture and track flow features and their statistics together with their correlation with relevant scalar quantities, *e.g.* temperature or species concentrations. In our approach we encode the set of all possible flow features by pre-computing merge trees augmented with attributes, such as statistical moments of various scalar fields, *e.g.* temperature, as well as length-scales computed via spectral analysis. The computation is performed in an efficient streaming manner in a pre-processing step and results in a collection of meta-data that is orders of magnitude smaller than the original simulation data. This meta-data is sufficient to support a fully flexible and interactive analysis of the features, allowing for arbitrary thresholds, providing per-feature statistics, and creating various global diagnostics such as Cumulative Density Functions (CDFs), histograms, or time-series. We combine the analysis with a rendering of the features in a linked-view browser that enables scientists to interactively explore, visualize, and analyze the equivalent of one terabyte of simulation data. We highlight the utility of this new framework for combustion science; however, it is applicable to many other science domains.

Index Terms—Topology, Statistics, Data analysis, Data exploration, Visualization in Physical Sciences and Engineering, Multi-variate Data.

1 Introduction

Combustion provides the vast majority of the world's energy needs and in an effort to reduce our reliance on fossil fuels, there are significant programs underway in the combustion science community to predict efficiency and pollutant emissions for potential new fuel sources coupled with advanced combustor designs for propulsion and power. To make these assessments scientists use Direct Numerical Simulations (DNS) of turbulent flames [29], to study effects such as flame auto-ignition [19] and extinction [30]. One of the primary drivers of these phenomena is the rate of turbulent mixing, characterized locally by the scalar dissipation rate, χ. Compressive strain in directions aligned to scalar gradients, creates thin pancake-like regions in the simulation whose thickness provides a direct measure of the local mixing length-scale and its scaling with turbulence length scales. Furthermore, experimental evidence [32] suggests that thickness and mean temperature within these features are related. A more thorough understanding of this relationship would allow scientists to better characterize the effects of mechanical strain from turbulence on chemical processes and provide fundamental insights into the properties of turbulent flames.

However, this type of analysis poses several challenges: The scalar dissipation structures are typically defined using contours at locally varying isovalues [34]. Since a wide range of values produce plausible

- *J. C. Bennett, J. H. Chen, and J. Shepherd are with Sandia National Laboratories, Email: {jcbenne, jhchen, jfsheph}@sandia.gov.*
- *V. Krishnamoorthy, S. Liu, and V. Pascucci are with the Scientific Computing and Imaging Institute at the University of Utah, Email: {vaidy, shusenl, pascucci}@sci.utah.edu.*
- *R.W. Grout is with National Renewable Energy Laboratory, Email: ray.grout@nrel.gov.*
- *E.R. Hawkes is with the University of New South Wales, Email: evatt.hawkes@unsw.edu.au.*
- *P.-T. Bremer is with Lawrence Livermore National Laboratory, Email: bremer5@llnl.gov.*

Manuscript received 31 March 2011; accepted 1 August 2011; posted online 23 October 2011; mailed on 14 October 2011.
For information on obtaining reprints of this article, please send email to: tvcg@computer.org.

structures, a large number of different segmentations must be explored to determine the sensitivity of the results to changes in parameters or to find stable thresholds. Furthermore, scientists are interested in conditioning their analysis on additional parameters, such as temperature variance, introducing additional free parameters to be explored. Finally, many of the hypotheses are initially derived from visualizations of the temporal behavior of the flame. Thus, it is important to provide visual feedback on the impact parameter choices have on the nature of the χ structures. These challenges are exacerbated by the massive size of the simulation, which in this case is roughly one terabyte.

Historically, scientists have relied on conditional statistics, applied globally, to effectively reduce the data to manageable proportions. However, even advanced indexing schemes [45, 31] are restricted to queries based on either function ranges or pre-computed properties. As will be discussed below, regions of high χ cannot be extracted through range queries. Furthermore, while pre-computing a single set of structures is feasible, the appropriate parameter choices are not known *a priori* and extracting a large number of different sets for exploring the parameter space is infeasible given the data size. Finally, traditional statistics typically provide only global averages rather than per-feature information, making simple queries such as how many features exist overall, difficult to answer.

We have developed a new integrated analysis and visualization framework to support combustion research. Our system enables a free choice of feature parameters and conditional sub-selections and interactively produces a wide range of diagnostic plots equivalent to the processing of the entire data set. Furthermore, the statistics viewer is cross-linked to a visualization of the corresponding three dimensional structures, enabling selection of (sets of) features on either end. The feature visualization employs a specialized volume rendering technique optimized for sparse, dynamic, and binary segmented volumes.

Instead of extracting a single set of features, we compute a multi-resolution hierarchy, capable of representing features for different parameters and at various scales. In a single pass over the original data we pre-compute a large variety of statistics for the finest resolution features. At run time the user selects parameters resulting in a set of features whose properties are aggregated on-the-fly, allowing the user to explore an entire family of feature definitions without accessing the original data. By pre-computing statistics for a base set of features, and providing the user with several multi-resolution hierarchies to explore, our system provides significantly greater flexibility in the analysis process than the typical range queries of indexing schemes. Additionally, the run-time aggregation avoids much of the cost of re-computing statistics for each set of features. As a result, our approach delivers the flexibility of extract-and-analyze techniques while allowing for interactive exploration of large data on a commodity desktop machine. Our system has been deployed at the Combustion Research Facility at Sandia National Laboratories and is actively being used by combustion scientists to glean insight from state of the art combustion simulations. Our contributions in detail are:

- On-the-fly aggregation of feature-based spatial and temporal statistics for large-scale simulations;
- An efficient encoding method for various multi-resolution hierarchies and statistics using feature-based blocked storage;
- A system for interactive creation of spatial and temporal statistical summaries including conditional empirical Cumulative Distribution Functions (CDFs), histograms, time-series, and parameter studies;
- An interactive feature browser designed using a novel volume rendering technique; and
- A linked view system of statistics and features with an intuitive user interface for feature selection and highlighting.

To demonstrate the framework we use an analysis of the relationship between length-scales and temperature of regions of high χ in turbulent combustion simulations. However, the design and implementation of our tools are general and can be applied broadly in other scientific domains where feature-based analysis is relevant.

2 Turbulent Combustion

Combustion currently provides 85% of our nation's energy needs and will continue to be a predominant source of energy as fuel sources evolve away from traditional fossil fuels. Low emission, low-temperature engine concepts of the future operate in regimes where combustion is poorly understood. In an effort to reliably predict efficiency and pollutant emissions for new engines and fuels, computer simulations are used to study fundamental turbulence-chemistry interactions. Direct Numerical Simulations (DNS) are first principle, high-fidelity computational fluid dynamics simulations in which the reactive compressible Navier-Stokes equations are numerically solved on a computational mesh in which all of the spatial and temporal scales of the turbulence are resolved [15]. In many practical turbulent combustion situations, turbulence strains the flame, causing molecular mixing of reactant streams. With increased mixing, chemical reactions are enhanced and overall efficiency increases up to a point, at which the loss of heat and radicals exceeds their rate of generation due to chemical reaction and the flame extinguishes resulting in increased emissions. Heat-release caused by the chemical reactions creates a complex feedback mechanism, affecting the intensity of the turbulence through density and viscosity changes across the flame.

Turbulent mixing is characterized locally by the scalar dissipation rate, χ, the rate at which scalar fluctuations decay due to diffusive processes. Compressive[1] turbulent strains create thin pancake-like regions of locally high dissipation rate. The morphology of these features, characterized according to their first three length-scales: length, width and thickness, is assumed to be correlated with length scales of turbulence. The thickness is particularly relevant as it provides a direct measure of the local mixing length-scale. Understanding the relationship between the thickness and the mean temperature within the features is of principal interest in order to study the relationship between mechanical strain and chemical processes.

There is experimental evidence [32], that the χ-layer thickness distributions are self-similar as $(T/T_0)^n$. In the measurements of Frank and Kaiser [32], it was determined that $n \approx 0.75, T_0 \approx 400$k provided an optimal collapse of the thickness Probability Density Functions (PDFs) conditional on various temperatures. In this paper, we use our framework to extract the thickness PDFs and determine if the same scaling is valid for the DNS data considered here.

3 Related work

Analysis of χ structures: Previous DNS and experimental results [25, 32, 42] have shown that turbulent strain in a non-reactive jet or shear layer leads to regions of intense mixing rates, which are oriented by the directions of principal strain rates, characterized by relatively large dimensions in the tangent plane of the principal strain rates, and a much smaller dimension in the direction normal to the principal strain rate. Reactive flows have been studied more recently [30, 34] by directly computing and tracking χ within the simulation. However, measurements were limited to thickness only and did not explore the relationship between regions of high χ and temperature.

Data warehouse technologies: At its core, our system relies on fast and efficient statistical queries for scientific data. Assuming entirely pre-computed information this problem reduces to finding and aggregating data from a large collection of records. This is a common challenge typically addressed by large Database Management Systems (DBMSs) [16]. In such systems, each feature would be represented as one record with its corresponding statistical information collected as entries in the record. In addition to the raw data, DBMSs compute multi-dimensional search structures such as B+-trees [18] that provide efficient searches for sub-selection type queries. However, traditionally DBMSs are designed to support constantly changing information, *e.g.* bank transactions, and thus their index structures have to trade query efficiency for the ability to change indices on-the-fly. On the

[1] By the term 'compressive', we mean a strain rate tensor which is compressive of scalar iso-surfaces when projected into the direction normal to the iso-surface.

contrary, scientific data is typically computed once and updated rarely if ever. This allows more efficient data management relying on static indices. To distinguish such systems from DBMSs they are typically referred to as data warehouses [23, 31, 14].

One particularly successful data warehouse technology is the FastBit system [45]. Instead of search trees, FastBit relies on compressed bitmask indices [46, 47] to provide efficient subselections and can significantly out-perform other approaches [37, 40]. However, this class of data management techniques relies almost exclusively on extracting and aggregating pre-computed information. As a result, data warehouses are well suited to access information computed for one particular set of features. In an exploratory setting, however, when the exact feature definition is unknown, warehouses are often too inflexible. While it is possible to pre-compute statistics for multiple sets of features, this is computationally expensive and the system remains limited to a small number of pre-defined feature sets. Instead, our framework uses a general feature hierarchy that allows the user to interactively change the parameters defining the features and thus explore an entire family of feature sets. Considering the inherent flexibility of common multi-resolution hierarchies, pre-computing statistics for all possible combinations of features is infeasible.

Statistics: To avoid excessive pre-processing, we leverage recent developments in parallel statistical algorithms [6, 35] to quickly aggregate first through fourth order moments. With recent increases in data sizes there have been a number of efforts to develop robust, parallel and/or streaming statistics algorithms. Of particular interest are the centered moments and co-moments which are the building blocks of many algorithms. In [44] a single-pass algorithm for the computation of variance was developed. A more general set of pairwise update formulas for variance was introduced in [12, 13]. The formulas for third- and fourth-order moments, which are needed to calculate skewness and kurtosis of the data set, were derived by [41]. Numerically stable, single-pass update formulas for arbitrary centered statistical moments and co-moments are presented in [6, 35]. There also exist a number of commercial packages such as MatLab [1] and SAS [3] that support parallel statistics. By coupling, the pair-wise update formulas developed for parallel statistics computation with a general feature hierarchy, our framework provides interactive exploration of feature-based statistics.

Feature hierarchies: As discussed in Section 4, our system implements a general multi-resolution hierarchy of features [21] in which a certain number of initial high-resolution features are combined according to a scale parameter. Of particular interest in this context are a number of topology-based hierarchies proposed in various settings and using a diverse set of algorithms. Using a variety of metrics such as feature volume, hyper-volume, or persistence, Carr et al. [11] use hierarchical contour trees [10] to define anatomical structures. Isovalue-dependent statistics are introduced in [5], where surface area, volume, and gradient integral of contours are plotted to provide the user quantitative measures for parameter selection. Using persistence based hierarchical Morse complexes, Laney et al. [33] show how different "resolutions" of the Morse complex encode progressively coarser segmentations of bubbles in Rayleigh-Taylor instabilities. Other examples include threshold-based hierarchies for non-premixed [34] and premixed [7] combustion simultations and core structures in porous media [26]. These topological techniques are particularly attractive for feature-based statistics as their hierarchical structure allows for features to be defined by, for example, varying isosurface thresholds. In general, other types of feature-based techniques that provide threshold-dependent feature definitions [39, 9] or clustering methods [28] could be suitable as well.

4 Feature-Based, Statistical Analysis

The framework described in this paper is based on two linked components: Fast creation of feature-based statistics and an interactive display of the corresponding feature geometry. This section describes the general structure of a feature-based hierarchy, the specific hierarchy used in the case study, as well as the run-time system for the creation of statistical summary plots.

4.1 Augmented Feature Families

One of the basic concepts of our framework is the notion of a *feature family*. Given an algorithm to define and extract features of interest corresponding to a parameter p, a feature family is a one-parameter family that for every possible parameter p stores the corresponding set of features. While any feature definition can be used to create a feature family by exhaustively pre-computing all possible features for all possible parameters, many popular algorithms naturally produce nested sets of features for varying parameters. For example, clustering techniques progressively merge elements [38, 17] and a threshold-based segmentation creates increasingly larger regions [8]. In these examples all features can be described by a collection of base elements (*e.g.* clusters) or as a collection of differences between features at different parameters (*e.g.* regions above threshold a that are below threshold b) respectively.

Feature families with a nested structure can be encoded and computed in an efficient manner. In our system, we specify for each *element* in the hierarchy its *life span* (in terms of the feature parameter), an arbitrary number of *children*, and a single *parent*. As is common with hierarchies, the set of *features* at a particular parameter p is then defined as all elements that are *alive* at parameter p combined with all their decendents. More formally we define:

Definition 1 (Element). *An* element *e is defined by a unique id and minimally contains a parameter range* $[p_{min}, p_{max}]$*, a direction, a collection of its children ids, and the id of its parent:*

$$e = (id, direction, [p_{min}, p_{max}], \{child_0, \ldots, child_n\}, parent) \in \mathbb{E}$$

Definition 2 (Feature). *A* feature *f is the union of an element e and all its descendents*

$$f = \{e \cup children^n(e) | n \in \{1, 2, ...\}\}$$

The element *id* is simply a unique identifier that is typically stored implicitly, *e.g.* based on the order in which elements are stored in a file. The direction indicates whether the parent of an element is *born* at $p < p_{min}$ and consequently its children are born at $p > p_{max}$ or the opposite.

A feature family is a collection of features defined hierarchically as described above:

Definition 3 (Feature Family). *A* feature family *F is a set of features*

$$F = \{f_0, \ldots, f_m\} \subset \mathbb{F}$$

Finally, in a time-dependent simulation or an ensemble of simulations we have one feature family per time or ensemble member:

Definition 4 (Clan). *A* clan *C is an ordered set of feature families*

$$C = \{F_0, \ldots, F_n\} \subset \mathscr{F}$$

We store feature families in a traditional multi-resolution graph that is updated on-the-fly as the user changes parameter. At any time we maintain a set of living elements that serve as the representatives for their corresponding features. Using the parent and child information this set is progressively updated as the feature parameter changes. Specifically, when an element dies it is removed from the set and either its children or its parent are born and added to the set. Furthermore, we support the encoding of multiple hierarchies associated with a feature family by storing multiple parameter ranges and child/parent ids in each feature, one for each hierarchy. In this particular case study we define feature families using merge trees with either relevance- or threshold-based segmentations.

Merge Tree Based Feature Families. As discussed above, the features of interest are regions of locally high χ. As shown in [34, 8] the *merge tree* is ideally suited to hierarchically encode such regions.

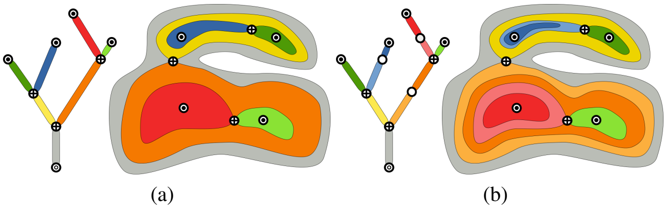


Fig. 2. (a) Merge trees represent the merging of contours as a function is lowered through its range. Each branch represents a portion of the domain as indicated by the colors. (b) To increase the resolution in parameter space we refine the merge tree by splitting long branches and refining the segmentation accordingly.

Given a simply connected domain $\mathbb{M}$ and a function $g : \mathbb{M} \to \mathbb{R}$ the *level set* $L(s)$ of g at isovalue s is defined as the collection of all points on $\mathbb{R}$ with function value equal to s: $L(s) = \{p \in \mathbb{M} | g(p) = s\}$. A connected component of a level set is called a *contour*. The merge tree of g represents the merging of contours as the isovalue s is swept top-to-bottom through the range of g, see Figure 2(a). Each branch of the tree represents a family of contours that continuously evolve without merging as s is lowered. These contours sweep out a subset of $\mathbb{M}$ and thus the branches correspond to a segmentation of $\mathbb{M}$, see Figure 2(a). To increase the resolution in parameter space we refine the merge tree by splitting long branches and refining the segmentation accordingly, see Figure 2(b).

In a simple threshold-based segmentation, each branch of the tree is an element with a lifetime given by the function values of the branch's top and bottom nodes. Given a particular threshold, each branch acts as the representative of its subtree/feature and, by construction, each subtree represents a simply connected region of high threshold, see Figure 3(a). However, when g spans multiple orders of magnitude *relevance* [34] is an alternate metric that scales g at each node by its local maximum – the highest maximum in its corresponding subtree. The relevance lifetime of a branch is thus given by the relevance interval between its top and bottom node and ranges between 0 and 1, see Figure 3(b). We compute merge trees and their corresponding segmentation using the streaming algorithm proposed in [8]. The input is a collection of vertices with function values, edges connecting them, and finalization information indicating when a vertex is no longer used. As the algorithm processes vertices, it maintains an active merge tree using a simple update procedure. By aggressively removing portions of the tree whose vertices have been finalized, the algorithm is fast while keeping a low memory footprint. In particular, the algorithm allows for the pre-screening of vertices with function values outside of a range of interest (in this case study very low χ values) and for the interleaving of file I/O with computation. However, apart from memory and efficiency concerns any other merge tree or contour tree algorithm could be used. For example, the publicly available libtourtre library [24] provides the necessary functionality. The output required by the downstream tools is a merge tree that for each branch contains a list of domain vertices that belong to its corresponding contours.

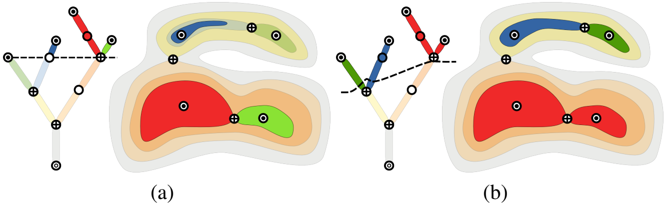


Fig. 3. (a) A threshold based segmentation of a merge tree at a threshold slightly above 80% of the global maximum. (b) A relevance based segmentation at relevance around slightly above 0.2 (slightly below 80% of the local maximum per branch). All local maxima are included and regions of higher function value (red) span a larger range.

Feature Attributes. In addition to the information necessary to encode a feature family we augment each feature with an arbitrary number, k, of additional attributes $(att^0, \ldots, att^k)$. Our system currently supports various descriptive statistics such as minima, maxima, first through fourth order statistical moments and sums, as well as as shape descriptors such as volumes and various length-scales. Descriptive statistics are computed incrementally as the feature family is constructed, using the same update formulas [6, 35] employed for the interactive aggregation during data exploration (Section 4.2). Specifically, as each vertex is labeld with its branch id, the vertex's associated attributes are added to the corresponding statistical aggregator. While this incremental approach works well for descriptive statistics, certain attributes such as shape descriptors cannot easily be computed in this manner, and are thus computed in a post-processing step. As discussed above, each element stores a list of corresponding vertices making it straight forward to assemble all vertices of a feature. Given this set of vertices we estimate the first three length-scales (length, width, and thickness) using a spectral technique similar to the one introduced by [36]. First, we compute a boundary surface of the vertex set as an iso-surface of a binary segmented volume. We then parametrize this shape according to its first non-trivial eigenvector to compute its length (Figure 4(a)). Subsequently, we extract iso-contours of this parametrization and apply the same technique recusively to compute the width (Figure 4(b)) and once again for the thickness (Figure 4(c)). While we typically compute descriptive statistics during merge tree construction, they could also be computed as a post-process given the list of vertices and their attributes. Doing so would allow the use of traditional, multi-pass algorithms, but would require all attributes to be accessible during the post-process, resulting in additional file I/O.

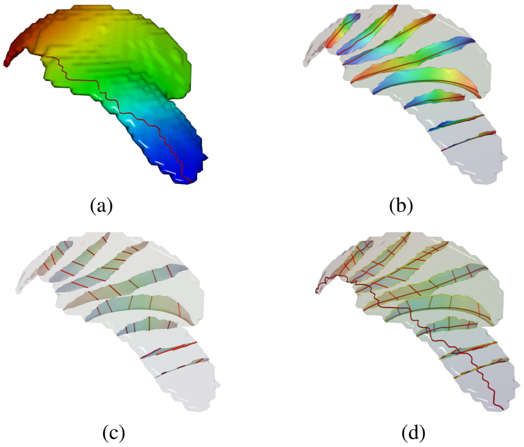


Fig. 4. The first three length-scales of a feature are estimated using a spectral technique. Each shape is parametrized according to its first non-trivial eigenvector to compute its length (a), and the same technique is performed recursively on iso-contours of the first eigenvector to compute the width (b) and thickness (c).

4.2 Interactive Exploration of Feature-Based Statistics

One of the main advantages of our system is the ability to quickly explore a wide variety of statistical information based on the given feature definitions. To achieve this our framework supports four operators that map feature families, sets of features, and statistics into new sets of features, or scalar quantities:

Definition 5 (Selection). *A selection* $S : \mathscr{F} \times \mathbb{R} \to \mathscr{P}(\mathbb{F})$ *is an operator that, given a feature family and a parameter, returns a set of features as well as (a subset) of their corresponding attributes.*

Note that each feature stores attribute information regarding the portion of the domain it covers, see Figure 2(a). A selection will, for most attributes, aggregate all values in the associated subtree on-the-fly as the hierarchy is navigated. This preserves the flexibility to base

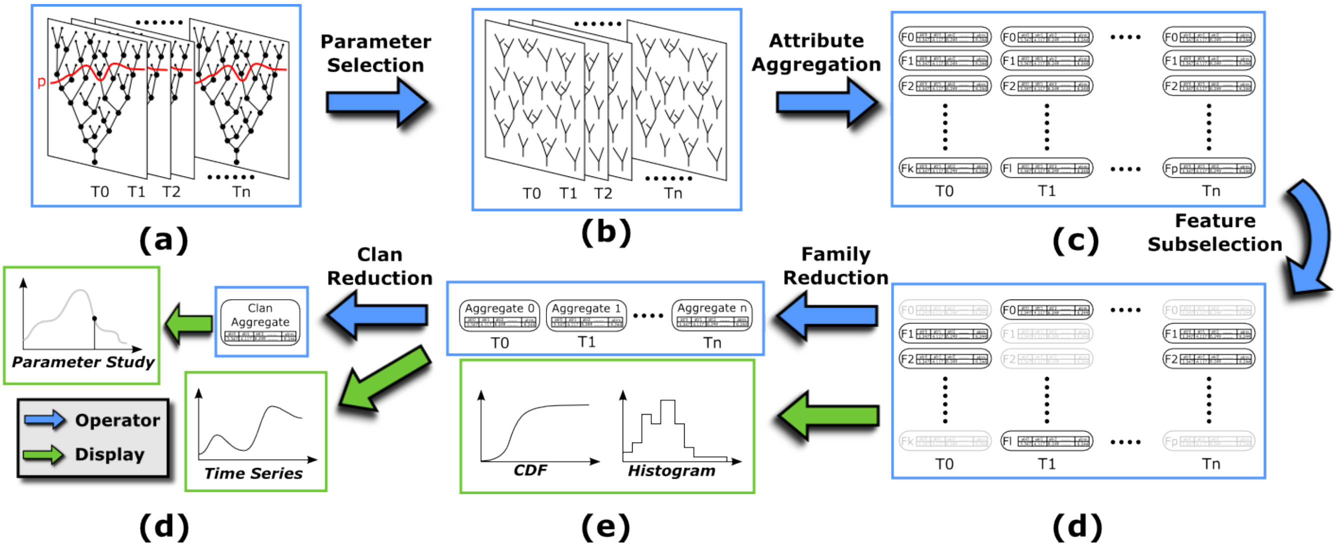


Fig. 5. Computational pipeline for interactive feature exploration. Starting from a clan of feature families represented by a sequence of merge trees (a) setting the feature parameter results in a sequence of sets of features each represented by a subtree of elements (b). Aggregating statistical attributes for each feature produces a set of features with attributes for each time step (c). A subselection on an arbitrary attribute narrows this collection to features of interest (d). Subsequently, either clan wide plots such as CDFs are created (e, bottom) or a reduction operator is applied to each family to create a time series of aggregated attributes (e, top). Finally, the time series is plotted (f, bottom) or an additional reduction is used to create a clan wide aggregated scalar property (f, top), which produces a single sample of a parameter study. A full study is created by repeatedly executing the pipeline.

different feature families on the same set of initial attributes. Nevertheless, if only one type of family is needed, aggregation of attributes can be performed once and stored to accelerate the exploration, see Section 4.3.

Definition 6 (Aggregation). *An* aggregation $A : \mathscr{P}(\mathbb{F}) \times \{0,...,k\} \to \mathbb{R}$ *is an operator that, given a set of features and an attribute index, returns the combined attribute for the set of features.*

Definition 7 (Subselection). *A* subselection $U : \mathscr{P}(\mathbb{F}) \times \{0,...,k\} \times \mathbb{R}^2 \to \mathscr{P}(\mathbb{F})$ *is an operator that, given a set of features, an attribute index, and a corresponding attribute interval range, returns the subset of features whose attribute value is contained in the interval.*

The subselection operator facilitates the creation of conditional plots, which are often an important part of the analysis pipeline.

Definition 8 (Reduction). *A* reduction $R : \mathscr{P}(\mathbb{R}) \to \mathbb{R}$ *is an operator that given a set of scalar values returns a single scalar value, for example by computing the mean.*

Using the operators described above we create three different types of plots as summarized by Figure 5: species distributions, parameter studies, and time-series. To simplify the discussion below, we assume that the input to each of the operators is all feature families in a clan, even though in practice we support the restriction to subsets of the data.

All plots take as input a feature clan C, a parameter p, subselections $Q = \{(att^{i_0}_{min}, att^{i_0}_{max}),\ldots,(att^{i_l}_{min}, att^{i_l}_{max})\}$, and an attribute index i. First, the parameter p is used to select an initial set of features from the clan, which are then further subselected using the subselections Q.

Species distributions plots include histograms and empirical CDFs, and track the distribution of the attribute att^i. For example, a major focus of our case study is the distribution of the mean thickness of χ structures conditioned on both the variance and mean of temperature within each feature, see Figure 6(a).

A time-series, as the name suggests, shows the evolution of att^i over time, and requires an additional family-wide reduction operator, R_f, as input. For example, one might plot the maximum temperature variance of features over time. In this example R_f is the maximum of att^i, which is temperature variance, see Figure 6(b).

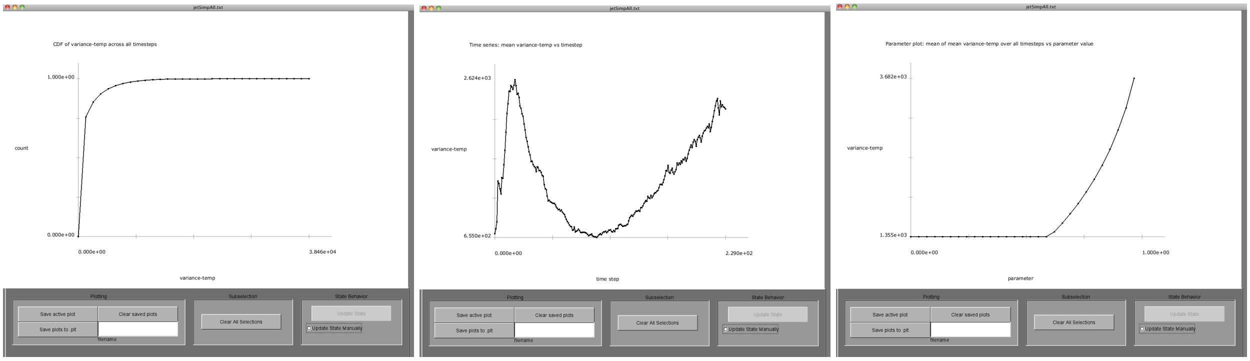
Parameter studies are an extension of time-series that show how att^i changes as the parameter p is varied. For these plots a clan-wide reduction operator, R_c, is required in addition to R_f. Extending our previous example, one might plot the mean of the maximum temperature variance of features as parameter p varies. In this example R_f is maximum of the temperature variance across time steps, and R_c is the mean of these values. Note that parameter studies can be come expensive as the range and granularity of p increases, because attributes must be aggregated for each p-value independently, see Figure 6(c). While parameter plots are the most expensive to produce they are also often very useful. In particular, a parameter plot shows how stable or unstable a given analysis is to the parameter selection. This is crucial in any exploratory setting to guarantee that the basis of important conclusions is not an inherently unstable analysis.

We provide a convenient GUI that allows the user to specify which attributes they would like to explore, loading only those to minimize memory overhead. Subselection sliders are generated for each specified attribute automatically and, if multiple hierarchies are available, the user can toggle between these and can update parameters interactively. Optional log scaling is provided, and radio buttons are used for selection of family- and clan-wide reduction operators.

The plot viewer is linked to the feature browser described in Section 5 to provide context as statistics are explored. Only those features that have been subselected using the GUI sliders are displayed by the feature browser. Users can click on an individual feature in the feature browser to obtain details on its associated statistics. Furthermore, when the user picks regions of histograms or CDFs, only those features that are contained in the selected bins are displayed by the feature browser, see Figure 1 for an example.

4.3 File Format

We store feature families and their corresponding attributes in a modular and easily extendable file format. Typically, we save one file per feature family to easily allow the restriction to temporal subsets, for example. At the end of each file we store an XML-footer followed by the file offset to the start of the footer as the last eight bytes in the file. The XML structure encodes which components are stored for the feature family, and typically comprises a simplification sequence storing the hierarchy information in addition to a number of attributes.



(a) (b) (c)

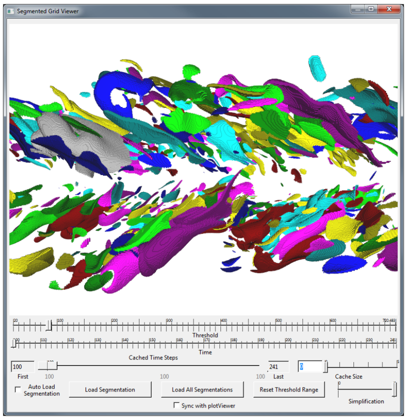
Fig. 6. Our system supports three types of plot generation: (a) species distribution, (b) time-series, and (c) parameter studies.

Any attributes stored indicate their type in addition to meta-data such as the name of the source field, how many bytes are used for each value, and whether data is stored in binary or ascii format. For the statistical moments we store not only the final value, *e.g.* mean, but enough information to further aggregate multiple values as needed by the parallel statistics formulas of [6, 35]. This requires each n-th order statistical moment to store all lower-order moments to support aggregation. Most importantly the XML structure stores file offsets to each corresponding block of data, allowing for the selective loading of subsets of attributes for exploration. One immediate advantage of this file structure is that it can be easily extended without re-writing entire files. Given a new set of attributes, we read the XML footer, append the new data at the end of the old data (overwriting the old footer), update the footer, and append it to the file.

5 Interactive Feature Browser

The previous section discussed how to extend, and accelerate traditional statistical analysis using feature-based hierarchies. In this section we introduce an interactive display of the segmentations corresponding to feature families, enabling the exploration of large time-series or ensembles in a feature driven manner. Once a feature definition has been identified and precomputed, the user can interactively adapt the segmentation to any given parameter. While general statistical and visualization tools, such as **R** [2] and **VTK** [4], are available our framework provides unique capabilities with respect to a given set of feature definitions. For example, in our case study, χ structures are extracted as local isosurfaces, see Figure 7. While isosurfacing is a standard procedure, using localized thresholds is not. Furthermore, even when using a global threshold, separating an isosurface on-the-fly into individual connected components for per-feature selection and display is a non-standard capability.

At its core, the interactive display is based on the same hierarchy used for the statistical analysis. However, in addition to the feature family we also store the corresponding segmentations in the form of a separate file containing for each feature a list of indices. These encode vertex locations with respect to a regular grid. Typically, features are sparse within a data set and, depending on the application, we have experienced no more than 3-4% of the domain being part of features. Additionally, given the sparse nature we can easily accommodate very large regular grids as well as AMR grids. For the latter we simply store samples relative to a virtual grid at the smallest scale present in the file.



First: 100, Last: 241, Cache Size: 0

□ Auto Load Segmentation
Load Segmentation
Load All Segmentations
Reset Threshold Range
□ Sync with plotViewer

Fig. 7. To support efficient volume rendering of dynamic feature hierachies, we combine a recent technique for octree-based GPU raytracing with a segmented id filtering scheme and extend the resulting system to dynamically changing volumes.

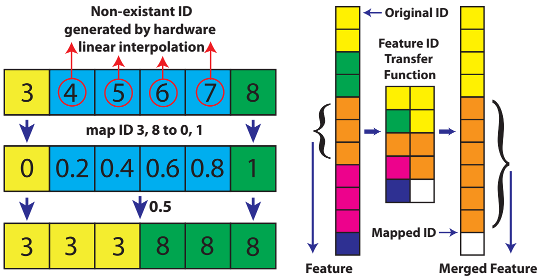
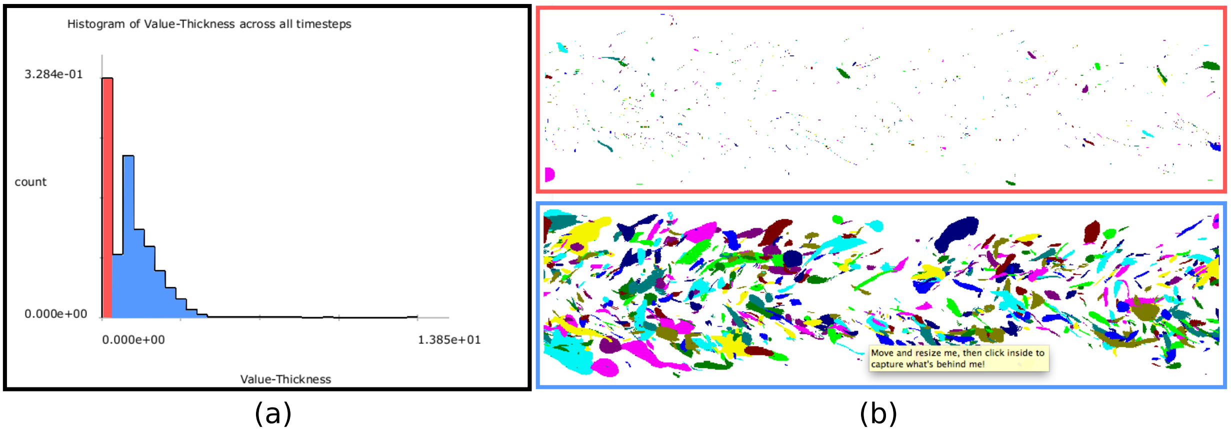


Fig. 8. The diagram on the left demonstrates interpolation between features in the GPU. Fragments are represented by rectangular blocks, and blue blocks have interpolated values. The first row shows that direct interpolation will reference incorrect feature ids. By instead mapping the neighboring feature id to 0-1, correct per-fragment filtering and classification is achieved. The diagram on the right illustrates our transfer function used to mitigate the cost of reloading element ids to the GPU. Rectangular blocks represent stored element ids (colors indicate different id numbers, while white represents empty space). The left column shows the original element id volume, while the right column is the result of applying the transfer function showed in the middle.

Similar to the statistics computation, the geometry of a living feature is defined as the collection of samples of its representative element combined with those of its descendants. Storing only indices rather than original data values (*e.g.* χ values) reduces the data overhead and provides significant flexibility in the type of source data the system can represent natively. However, as a result, a segmented feature family represents a family of “binary” segmented data with integer indices at each grid point. This presents a challenge for rendering since (a) descendants must be drawn in the same color as their representatives and (b) no simple interpolation to smooth and appropriately light the boundary of features is available.

We combine recent techniques for octree based GPU raytracing [20] with a two-level volume rendering approach [27] and extend the result-



Move and resize me, then click inside to capture what's behind me!

Fig. 9. (a) The unscaled histogram of thicknesses for χ structures with mean temperature between 829 and 1078. A large number of features populate the first bin (virtually independently of the bin sizes). (b) Selecting only structures falling within the first bin contains predominantely artifacts of the segmentation (top), while the remaining bins contain almost exclusively features of the expected characteristics (bottom).

ing system to dynamically changing volumes. Given the sparseness of the input features, using an adaptive octree to store feature samples allows us to significantly reduce the memory footprint compared to storing a regular grid. In our experience we use between 2% and 20% of the memory that a regular grid would require depending on the feature density. Furthermore, the octree structure facilitates highly effective empty space skipping, enabling ray marching with shorter marching length, thereby drastically reducing rendering time.

The second challenge is that neighboring elements often have widely different ids making interpolation difficult. As shown on the left of Figure 8, given two ids (3 and 8), a direct interpolation would include multiple inapplicable element ids. We solve this problem by using a 0-1 mapping approach, as described in [27], that was also used successfully to identify path tube points of simplified topological structures in [43]. Adjacent voxels with different ids are mapped to 0 (the smaller id) and 1 (the larger id), and then tri-linear interpolation in the 0-1 range is preformed using a shader program. Subsequently, we classify the areas with value greater than 0.5 as belonging to the larger id and areas with value less than 0.5 as belonging to the smaller id with per-fragment precision.

Finally, as discussed above, each feature is typically represented by a set of ids consisting of the id of its representative feature as well as the ids of all its descendants. Changing the feature parameter changes the composition of these sets and thus the segmentation. From the rendering engine's perspective, the volume is constantly changing as the hierarchy is explored. To mitigate the unnecessary cost of reloading the feature id volume into the GPU, we introduce a special purpose feature id transfer function generated directly from the feature family. As shown on the right of Figure 8, the column on the left represents the original element id volume, while the column on the right represents the mapped element ids. The transfer function maps the id of a dead element to empty space, that of a living element to itself, and that of a merged element to the id of its representative. To support a large number of elements, we use a texture buffer object in OpenGL to store the transfer function, capable of supporting a 1-dimensional lookup table as big as 2^{27} many entries.

By connecting the interactive segmentation display with on-demand statistical analysis, our framework provides new capabilities which have significantly accelerated the time to insight is our case study. By concentrating on those aspects of statistical analysis and visualization most useful to scientists, we have created an integrated framework which combines the traditional exploration and analysis cycles into a single interactive system. The user can simultaneously create plots to accept or reject specific hypotheses while visually exploring the data. Through the linked selection we provide the ability to quickly understand which features are responsible for which aspects of, for example, a CDF and conversely, picking interesting features provides information on all their relevant attributes.

6 Results

In the case study presented here the data describes a temporally-evolving turbulent CO/H_2 jet flame undergoing extinction and reignition at different Reynolds numbers [30]. The simulations were performed with up to 0.5 billion grid points and periodic boundary conditions in the mean flow (x) direction. The periodicity causes mixing rates to increase until approximately midway through the simulation, after which they begin to decay. Figure 10 shows a volume rendering of the χ field for one of the 230 timesteps saved for postprocessing analyses. In particular, for the example presented here we process three simulation variables, χ, mixture fraction, and temperature, representing just under one terabyte of raw floating point data.

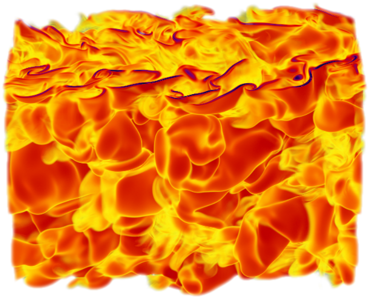
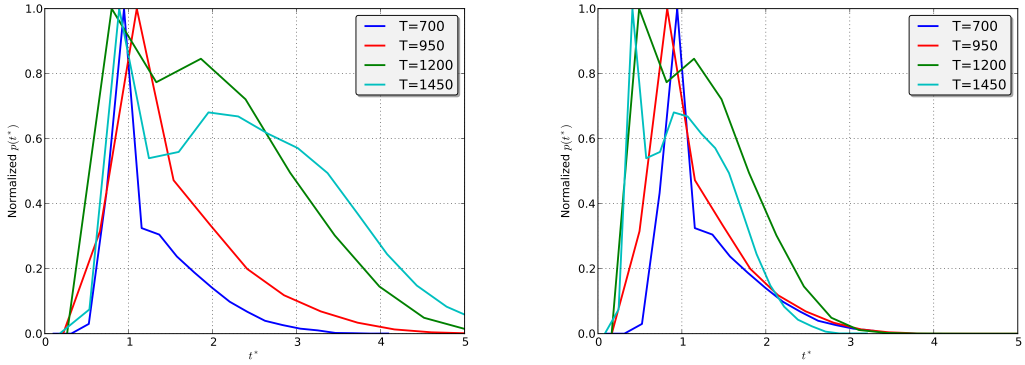


Fig. 10. The χ field from the temporally-developing CO/H_2 jet flame.

The primary scientific insights are captured in the distributions of Figure 11. In particular, Figure 11(a) shows the distribution of χ-thicknesses, at a relevance of 0.85, computed for structures grouped by the mean temperature in the segment for four bins, each 250 Kelvin wide. To ensure that the results are not influenced by excessive internal temperature variations, only structures with a low temperature variance (below 5% of the maximum variance) are considered. The flexibility to easily and interactively add such restrictions is a key feature of our framework. Furthermore, we detect several extremely small features which are likely artifacts of the segmentation. To mitigate the impact of such artifacts, which get collected in the first bin of the histogram, the first bin in the PDFs is discarded. The remaining conditional PDFs show a trend consistent with experimental observations: the thickness distribution conditional on temperature is asymmetric, with faster rise and shorter tail than lognormal, and shifts towards larger thickness with a broader distribution with increasing temperature. The restriction to segments of uniform temperature limits the



(a)

(b)

Fig. 11. (a) The distribution of the χ-thicknesses are computed for segments grouped by the mean temperature in the segment. (b) In order to compare our simulated results to previous experimental results, we apply the temperature scaling observed by Frank and Kaiser [32] to the conditional PDFs, by rescaling the abscissa according to:$t^* = t\,(T/T_0)^{-n}$ with $n = 0.75$ and $T_0 = 400$ and normalizing the conditional PDFs by their peak value.

sample size from which the distribution is drawn. Although each bin contains more than 300-10000 samples (depending on the conditional temperature band), the exact shape of the PDF is not easy to discern.

In Figure 11(b), we apply the temperature scaling observed by Frank and Kaiser [32] to the conditional PDFs, by rescaling the abscissa according to:

$$t^* = t\left(\frac{T}{T_0}\right)^{-n} \qquad (1)$$

with $n = 0.75$ and $T_0 = 400$ and normalizing the conditional PDFs by their peak value. As with the experimentally measured conditions, this value of n is effective in collapsing the PDFs, yet far smaller than would be expected from the dependence of the Batchelor length scale (a measure of the smallest scalar lengths expected in a non-reacting turbulent flow) on temperature through the kinematic viscosity (discussed in [32]). From the viscosity model used in the DNS, Batchelor scaling would suggest $n = 1.42$, almost twice the value found in both experiments and simulations.

This analysis demonstrates the power of our framework to easily and efficiently analyze and explore large-scale simulations on commodity hardware. Using the combined visual exploration and analysis capabilities, scientists were able to quickly produce a number of interesting statistics and immediately diagnose problems. For example, including the first bin in the PDF creates a large spike distorting the shape of the distribution, see Figure 9(a). However, examining the shapes of structures in the first bin, see Figure 9(b,top), it is easy to see that a large number are due to sampling and other artifacts. In a traditional setting, creating the unrestricted PDFs would have required another round of processing. The shape of the PDF would likely have caused concern resulting in a round of visualization to determine likely causes. Finally, at least one additional round of processing would be necessary to achieve the desired result shown in Figure 9(b,bottom). Given that each step involves one terabyte of data the file I/O alone would require several hours or large-scale parallel resources. Instead our framework provides this capability interactively on a commodity desktop.

Computing the merge trees, including the segmentations and descriptive statistics, took approximately five minute per time step depending on the progression of the simulation. Adding the length scales required an additional 90 minutes per feature family. All computations were performed in parallel using Oak Ridge National Laboratory's Lens system, a 32 node Linux cluster whose nodes comprise four quad-core 2.3 GHz AMD Opteron processors with 64 GB of memory, and 2 NVIDIA 8800 GTX GPUs. While no exact analysis parameters are known *a priori* and indeed parameter exploration is one of the crucial aspects in any analysis, in general scientists can provide conservative bounds on parameter values. In this case study, features of interest are regions of high χ, thus we ignore χ values below 20 since these are far below any region of interest. Similarly, the expected relevance values of interest lay around 0.8 and thus we pre-aggregated features below 0.6 to reduce file sizes without impacting the analysis. Finally, we also compute length scales only for a conservative range of relevance values between 0.6 and 0.9. Storing two feature families corresponding to the threshold and relevance, as well as mean and maximum χ values, temperature mean and variance, mean mixture fraction, and the first 3 length scales produces 945 MB of binary uncompressed data. This represents a 1000 fold data reduction without impacting the quality or flexibility of the analysis. The structure geometries require an additional 13 GB of binary, uncompressed index information.

Data exploration was performed on a single desktop machine with an Intel Core i7 2600k with 16 GB of DDR3 RAM, and Nvidia GTX580 GPU. Our interactive feature browser achieves framerates of 12-25 frames per second (depending on the number of blocks in the view frustum). Generating species distribution plots and time-series took on the order of a second, while parameter studies took 35 seconds to generate. The parameter studies were generated using a plot resolution of 40, and since each sample in a parameter plot represents a full analysis of the entire data set, this represents a 40-fold analysis of one terabyte of data performed in 35 seconds.

Although in this paper we have focused primarily on DNS combustion simulations, the framework applies equally well to other applications. For example, Figure 12(a) shows the weighted cummulative volume distribution of an idealized pre-mixed combustion simuation [22] along side the corresponding segmentation. This essentially mirrors Figure 9(d) of [7], however we use a fully volumetric analysis rather than restricting it to a temperature iso-surface and the plots are created interactively rather than in a batch process. Figure 12(b) shows the average density variance for different density thresholds in a simulation of hydrogen under pressure. As the threshold is lowered the variance increases up to a breaking point, after which it rapidly falls. The corrresponding segmentation is generated using the approximate threshold of the peak variance. Somewhat surprisingly, the peak variance does not correspond to the point at which the individual surfaces begin to merge, rather there are still a large number of well separated features.

7 User Evaluation

During the the design and implementation of our framework we have continuously discussed desirable capabilities and potential use cases with several groups of collaborators involved either in this case study or similar ones [7, 8]. Unsurprisingly, we found that providing tradi-

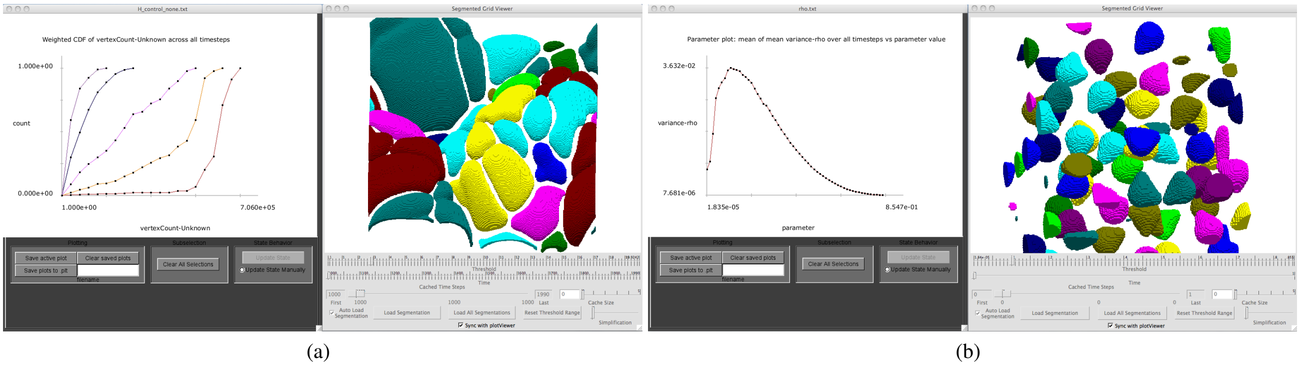


Fig. 12. (a) Weighted cumulative density function of feature volume in an idealized premixed hydrogen flame echoing Figure 9(d) in [7]. (b) The average density variance for features defined by different density thresholds in a simulation of hydrogen under pressure.

tional statistical plots is key to any practical analysis since these are the primary means by which scientific hypothesis are (dis-)proven and they provide a mechanism for the validation of new results and comparison with existing research. Certainly, there are pre-existing tools available to produce similar statistical summaries. However, we found several key capabilites that distinguish our design: First, the linked views and integrated selection capabilities provide immediate visual feedback on the impact of parameter choices and enable a quick yet highly detailed analysis, in particular of species distributions. Second, the ability to change feature parameters on-the-fly and interactively, provides a highly beneficial degree of flexibility. This relieves the user from guessing reasonable values before a potentially expensive and time consuming post-processing step and thus permits a much broader exploration of the problem space. Furthermore, the comparatively low investment in time and resources encourages the investigation of seemingly unlikey conjectures which often precedes new insights. Third, the fact that this exploration can be performed on common desktop systems rather than, for example, dedicated visualization clusters significantly lowers the barrier to entry. While parallel resources might be available, they are often cumbersome to use, involve a wait or start-up time, and rely on fast connection speeds. Instead, our framework performs well even on laptops and uses only very moderate amounts of disk space. This makes the analysis portable and convenient, an often overlooked yet practically important advantage.

In summary, the framework provides an intuitive GUI that enables traditional statistical distributions obtained from (un)conditional feature sets to be easily related back to the sets of features contributing to the statistic through feature-based visualization and linked views. Augmenting traditional statistical point-wise information are non-local geometrical statistics related to the size, shape, and proximity of relevant features. Moreover, statistical sensitivity to feature definitions - often conditional on multiple dependent variables whose threshold values are arbitrary and not known *a priori* - can be readily explored interactively within this framework irrespective of the overall data size. These attributes are essential as the datasets become larger, and more cumbersome to work with. Overall, the framework is a powerful tool for interactive exploration of large, multi-scale, multivariate, time-varying data sets, providing insights into the causality between turbulent flow structure and reactive processes.

In its current state, the framework takes simulation data as input and constructs augmented feature families to enable fast and efficient data exploration. However, a number of the attributes that are of interest are not raw simulation input data, rather they are derived quantities that must be post-processed by S3D. It would be advantageous if the framework incorporated S3D computation modules directly, for example the evaluation of chemistry, molecular diffusion, differentiation stencils, or thermodynamics, so as to provide a much richer, more relevant set of derived variables than is currently available. Furthermore, it would be advantageous if the framework provided the capability to compute the augmented feature families *in situ*, or in lock-step with S3D itself. This would require the efficient parallelization of merge trees in addition to coordination with native S3D data structures.

8 Conclusion and Future Work

We have presented a novel framework that combines topological and statistical analysis with visualization to perform feature-based statistical analysis of large scientific data. Our framework represents a novel technology that has converted the typically cumbersome post-processing cycle of explore and analyze into an interactive process, providing application scientists easy access to cutting edge visualization and feature-based analysis techniques coupled to traditional statistical techniques. In particular, this framework provides an intuitive GUI that enables traditional statistical distributions obtained from conditional feature sets to be easily related back to the sets of features contributing to the statistic through feature-based visualization and linked views. In this case study, our tool was successfully deployed in the Combustion Research Center at Sandia National Laboratories to gain insight into fundamental turbulence-chemistry interactions, demonstrating its practical application.

Going forward, we anticipate storing a much larger set of statistics per feature and/or providing more complex reduction operators. For example, given enough resources the length scale estimation could be applied at run-time to intermediate results. However, such a system would likely require small-scale parallel resources and a corresponding distributed analysis. Furthermore, we are exploring the use of this system for entirely different hierarchies based on for example, Morse-Smale complexes. This will likely result in different usage patterns and require additional capabilities. Finally, as we move towards exascale computing where I/O is severely limited, *in situ* processing of turbulence and combustion features, and partial parallel construction of distributed merge trees efficiently poses enormous challenges, particularly when the feature extraction algorithms must utilize native data structures from the distributed simulation.

Acknowledgments

The authors wish to thank David Thompson, Philippe Pébay and Ajith Mascarenhas for useful discsussions.

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under Contract DE-AC04-94AL85000.

This work was performed under the auspices of the US Department of Energy (DOE) by the Lawrence Livermore National Laboratory under Contract Nos. DE-AC52- 07NA27344, LLNL-JRNL-412904L.

References

[1] Matlab. http://www.mathworks.com/.
[2] R. http://cran.r-project.org/.
[3] SAS. http://www.sas.com/.
[4] VTK - The Visualization Toolkit. http://vtk.org/.

[5] C. L. Bajaj, V. Pascucci, and D. R. Schikore. The contour spectrum. In *IEEE Visualization '97*, Oct. 1997.

[6] J. Bennett, P. Pébay, D. Roe, and D. Thompson. Numerically stable, single-pass, parallel statistics algorithms. In *Proc. 2009 IEEE International Conference on Cluster Computing*, New Orleans, LA, Aug. 2009.

[7] P.-T. Bremer, G. Weber, V. Pascucci, M. Day, and J. Bell. Analyzing and tracking burning structures in lean premixed hydrogen flames. *IEEE Transactions on Visualization and Computer Graphics*, 16(2):248–260, 2010.

[8] P.-T. Bremer, G. H. Weber, V. Pascucci, M. Day, and J. B. Bell. Analyzing and tracking burning structures in lean premixed hydrogen flames. *Visualization and Computer Graphics, IEEE Transactions on*, 16(2):248–260, Mar.-Apr. 2010.

[9] H. Carr and J. Snoeyink. Path seeds and flexible isosurfaces - using topology for exploratory visualization. In C. D. H. G.-P. Bonneau, S. Hahmann, editor, *Proceedings of the symposium on Data visualisation*, pages 049–058, Grenoble, France, 2003. Eurographics Association.

[10] H. Carr, J. Snoeyink, and U. Axen. Computing contour trees in all dimensions. *Computational Geometry Theory Applications*, 24(3):75–94, 2003.

[11] H. Carr, J. Snoeyink, and M. van de Panne. Simplifying flexible isosurfaces using local geometric measures. In *IEEE Visualization '04*, pages 497–504. IEEE Computer Society, 2004.

[12] T. F. Chan, G. H. Golub, and R. J. LeVeque. Updating formulae and a pairwise algorithm for computing sample variances. Technical Report STAN-CS-79-773, Stanford University, Department of Computer Science, 1979.

[13] T. F. Chan, G. H. Golub, and R. J. LeVeque. Updating formulae and a pairwise algorithm for computing sample variances. In H. Caussinus et al., editors, *COMPSTAT 1982: 5th Symposium held at Toulouse, 1982*, pages 30–41, Wien, Austria, 1982. Physica-Verlag.

[14] S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. *ACM Special Interest Group on Management Of Data Record*, 26(1):65–74, 1997.

[15] J. H. Chen, A. Choudhary, B. de Supinski, M. DeVries, E. R. Hawkes, S. Klasky, W. K. Liao, K. L. Ma, J. Mellor-Crummey, N. Podhorszki, R. Sankaran, S. Shende, and C. S. Yoo. Terascale direct numerical simulations of turbulent combustion using S3D. *Computational Science & Discovery*, 2(1):015001, 2009.

[16] E. F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 19070.

[17] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:603–619, 2002.

[18] D. Comer. The ubiquitous B-tree. *Computing Surveys*, 11(2):121–137, 1979.

[19] D. Cook, H. Pitsch, J. Chen, and E. B. Hawkes. Flamelet-based modeling of auto-ignition with thermal inhomogeneities for application to hcci engines. *Proceedings of the Combustion Institute*, 31(2):2903 – 2911, 2007.

[20] C. Crassin, F. Neyret, S. Lefebvre, and E. Eisemann. Gigavoxels: ray-guided streaming for efficient and detailed voxel rendering. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games*, I3D '09, pages 15–22, New York, NY, USA, 2009. ACM.

[21] E. Danovaro, L. D. Floriani, P. Magillo, E. Puppo, and D. Sobrero. Level-of-detail for data analysis and exploration: A historical overview and some new perspectives. *Computers and Graphics*, 30(3):334–344, 2006.

[22] M. Day, J. Bell, P.-T. Bremer, V. Pascucci, V. Beckner, and M. Lijewski. Turbulence effects on cellular burning structures in lean premixed hydrogen flames. *Combustion and Flame*, 156:1035–1045, 2009.

[23] B. A. Devlin and P. T. Murphy. An architecture for a buisiness and information system. *IBM Systems Journal*, 27(1):60–80, 1988.

[24] S. Dillard. Libtourtre. http://graphics.cs.ucdavis.edu/∼sdillard/libtourtre/doc/html/.

[25] R. W. Grout, J. Chen, E. R. Hawkes, A. Mascarenhas, P.-T. Bremer, and V. Pascucci. Thirty-second international symposium on combustion, 2008.

[26] A. Gyulassy, M. Duchaineau, V. Natarajan, V. Pascucci, E. Bringa, A. Higginbotham, and B. Hamann. Topologically clean distance fields. *IEEE Transactions on Computer Graphics and Visualization (TVCG)*, 13(6):1432–1439, 2007.

[27] M. Hadwiger, C. Berger, and H. Hauser. High-quality two-level volume rendering of segmented data sets on consumer graphics hardware. In *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, VIS '03, pages 301–308, Washington, DC, USA, 2003. IEEE Computer Society.

[28] J. Hartigan. *Clustering Algorithms*. Wiley, 1975.

[29] E. Hawkes and J. Chen. Comparison of direct numerical simulation of lean premixed methane-air flames with strained laminar flame calculations. *Combustion and Flame*, 144:112–125, 2005.

[30] E. R. Hawkes, R. Sankaran, J. C. Sutherland, and J. H. Chen. Scalar mixing in direct numerical simulations of temporally evolving plane jet flames with skeletal CO/H_2 kinetics. *Proceedings of the Combustion Institute*, 31(1):1633 – 1640, 2007.

[31] W. H. Immon. The data warehouse and data mining. *Communications of the ACM*, 39(11):49–50, 1996.

[32] S. A. Kaiser and J. H. Frank. Imaging of dissipative structures in the near field of a turbulent non-premixed jet flame. *Proceedings of the Combustion Institute*, 31(1):1515–1523, 2007.

[33] D. Laney, P.-T. Bremer, A. Mascarenhas, P. Miller, and V. Pascucci. Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities. *IEEE Trans. Visualization and Computer Graphics (TVCG)*, 12(5):1052–1060, 2006.

[34] A. Mascarenhas, R. W. Grout, P.-T. Bremer, E. R. Hawkes, V. Pascucci, and J. H. Chen. *Proceedings of the 3rd TopoInVis Workshop*, chapter Topological feature extraction for comparison of terascale combustion simulation data, pages 229–240. Mathematics and Visualization. Springer, 2010.

[35] P. Pebay. Formulas for robust, one-pass parallel computation of covariances and arbitrary-order statistical moments. Technical Report SAND2008-6212, Sandia National Laboratories, 2008.

[36] M. Reuter, F.-E. Wolter, M. Shenton, and M. Niethammer. Laplace-beltrami eigenvalues and topological features of eigenfunctions for statistical shape analysis. *Computer-Aided Design*, 41(10):739–755, 2009.

[37] J. Schlosser and M. Rarey. Beyond the virtual screening paradigm: Structure-based searching for new lead compounds. *Journal of Chemical Information and Modeling*, 49(4):800–809, 2009.

[38] Y. Sheikh, E. Kahn, and T. Kanade. Mode-seeking by medoidshifts. In *Proc. IEEE International Conference on Computer Vision*, pages 1–8, 2007.

[39] D. Silver. Object-Oriented visualization. *IEEE Computer Graphics and Applications*, 15(3):54–62, May 1995.

[40] T. L. L. Siqueira, R. R. Ciferri, V. C. Times, and C. D. de Aguiar Ciferri. A spatial bitmap-based index for geographical data warehouses. In *Proc. SAC*, pages 1336–1342, 2009.

[41] T. B. Terriberry. Computing higher-order moments online, 2008. http://people.xiph.org/ tterribe/notes/homs.html.

[42] P. Vaishnavi, A. Kronenburg, and C. Pantano. On the spatial resolution for scalar dissipation measurement in turbulent jet flames. *J. Fluid Mech.*, 596:103–132, 2008.

[43] G. H. Weber, S. E. Dillard, H. Carr, V. Pascucci, and B. Hamann. Topology-controlled volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 13(2):330–341, Mar./Apr. 2007.

[44] B. P. Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420, 1962.

[45] K. Wu, S. Ahern, E. W. Bethel, J. Chen, H. Childs, E. Cormier-Michel, C. Geddes, J. Gu, H. Hagen, B. Hamann, W. Koegler, J. Laurent, J. Meredith, P. Messmer, E. Otoo, V. Perevoztchikov, A. Poskanzer, O. Ruebel, A. Shoshani, A. Sim, K. Stockinger, G. Weber, and W.-M. Zhang. Fastbit: Interactively searching massive data. *J. of Physics: Conference Series*, 180(1), 2009.

[46] K. Wu, E. Otoo, and A. Shoshani. A performance comparison of bitmap indexes. In *Proceedings of 10th International Conference on Information and Knowledge Management*, pages 559–561, 2001.

[47] K. Wu, E. Otoo, and A. Shoshani. Compressing bitmap indexes for faster search operations. In *Proceedings of Scientific and Statistical Database Management*, pages 99–108, 2002.