Statistical Visualization and Analysis of Large Data Using a Value-based Spatial Distribution

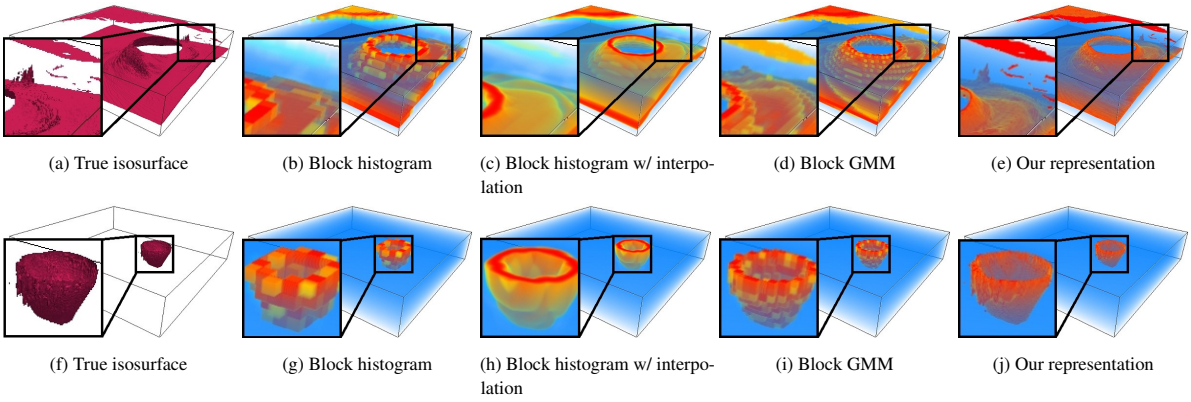
Ko-Chih Wang*
The Ohio State University

Kewei Lu†
The Ohio State University

Tzu-Hsuan Wei‡
The Ohio State University

Naeem Shareef§
The Ohio State University

Han-Wei Shen¶
The Ohio State University



(a) True isosurface (b) Block histogram (c) Block histogram w/ interpolation (d) Block GMM (e) Our representation

(f) True isosurface (g) Block histogram (h) Block histogram w/ interpolation (i) Block GMM (j) Our representation

Figure 1: Our *value-based spatial distribution* representation includes spatial information (called a Spatial GMM) that is compact, which is lacking in current distribution based representations. We illustrate our approach with volume rendering of the probability field for the location of an isosurface of the Isabel dataset (500x500x100). Figures (a) and (f) show the ground truth isosurface rendering using +90Pa and -900 Pa, respectively. Using a hot to cold transfer function, where red indicates a high probability for the isosurface and blue indicates a low probability, the rest of the columns show volume renderings of our approach (the last column) compared with current approaches. Equipped with added spatial information, our representation is able to better identify with higher certainty the location of the isosurface including small details, which are missed in the other representations. The block sizes used to compute the renderings in the last four columns are 12^3, 12^3, 6^3 and 16^3, respectively. In our approach, the value histogram costs 3.5MB and the Spatial GMM costs 3.26MB.

Abstract

The size of large-scale scientific datasets created from simulations and computed on modern supercomputers continues to grow at a fast pace. A daunting challenge is to analyze and visualize these intractable datasets on commodity hardware. A recent and promising area of research is to replace the dataset with a distribution based proxy representation that summarizes scalar information into a much reduced memory footprint. Proposed representations subdivide the dataset into local blocks, where each block holds important statistical information, such as a histogram. A key drawback is that a distribution representing the scalar values in a block lacks spatial information. This manifests itself as large errors in visualization algorithms. We present a novel statistically-based representation by augmenting the block-wise distribution based representation with location information, called a *value-based spatial distribution*. Information from both spatial and scalar spaces are combined using Bayes' rule to accurately estimate the data value at a given spatial location. The representation is compact using the Gaussian Mixture Model. We show that our approach is able to preserve important features in the data and alleviate uncertainty.

*e-mail: wang.3182@osu.edu
†e-mail:lu.321@buckeyemail.osu.edu
‡e-mail:wei.225@buckeyemail.osu.edu
§e-mail:shareef.1@osu.edu
¶e-mail:shen.94@osu.edu

2017 IEEE Pacific Visualization Symposium (PacificVis)
18-21 April, Seoul, Korea
978-1-5090-5738-2/17/$31.00 ©2017 IEEE

1 Introduction

The computational power of modern supercomputers allow scientists to model physical phenomena with high-resolution simulations. Data-driven analysis and visualization techniques help scientists understand the dataset with greater depth and create more precise prediction models. However, analyzing such large-scale scientific simulation data is challenging due to the incompatibility between memory limitations, I/O capacities, and high computational power [3, 19]. Reducing the size of the dataset is a viable option for visualization and analysis on commodity hardware, but information loss in the proxy representation must be controlled. Approaches to construct a proxy include lossy compression [10, 11, 32], subsampling [25], and statistical based data representations [2, 9, 20, 26]. The latter approach is increasingly becoming more popular not only for data reduction but also due to the ability to retain important statistical features.

Many statistically based data representations partition the data domain into non-overlapping block regions, where the scalar field within each block is represented by a statistical summary of the scalar field, e.g. the mean or standard deviation of the scalar values within the block. Storing a distribution of the scalar values, such as a histogram, better represents the scalar field than simple summary metrics, but still suffers from limited accuracy. Though recent work on uncertain volume rendering and isosurface extraction [1, 2, 14, 20, 26] has tried to address this issue, they have not tackled the

inherent problem of the loss of spatial information in the distribution. Thus, estimating the value at a position within a block using its corresponding distribution will incur inevitable approximation errors that are difficult to overcome.

We present a novel distribution based representation that incorporates spatial information at a small additional storage cost. The dataset is partitioned into blocks, where the data in each block is used to construct two types of distributions. One is a value distribution on the scalar values. The other is a spatial distributions where the locations of samples within the block are collected and stored as a multi-dimensional distribution for each value sub-range. Each multi-dimensional distribution is stored using a Gaussian Mixture Model (GMM) and hereafter referred to as a Spatial GMM. Whereas current approaches that use GMMs map data values to probabilities, our Spatial GMM maps the locations of the data points in different value ranges to probabilities. An adaptive scheme is employed to determine the number of Gaussian components that are required for each Spatial GMM, which makes the total size to store Spatial GMMs smaller than using a fixed number of Gaussian components. Given an arbitrary location, we utilize our representation to infer the probability for a value to reside at this location using Bayes' rule, which combines known information (the value distribution) and additional evidences (the Spatial GMMs) from a given condition. Equipped with this spatial information, our approach produces lower variance, and hence lower uncertainty, in the results of statistical based analysis and visualizations. Figure 1 shows an example of using our approach to provide clearer and more accurate uncertain isosurface rendering when compared with recently proposed distribution-based data representations.

The rest of the paper is organized as follows. Section 2 discusses related work and Section 3 provides an overview of our approach. Our representation is presented in Section 4. Section 5 discusses the use of Bayes' rule to integrate the Spatial GMM and the block distribution. Section 6 presents quantitative experiments using Root Mean Square Error (RMSE) to compare our representation with other distribution-based representations. We demonstrate in Section 7 that our approach provides a marked improvement in visual quality when applied to a variety of visualization and analysis tasks. Section 8 presents timings to construct and evaluate our representation as well influence of parameters and the scalability of our approach. Section 9 concludes the paper and discusses future work.

2 Related Work

Data representation for large data: A traditional and straightforward approach to reduce the size of volume data is to down-sample the volume to a smaller resolution. However, this approach suffers from aliasing and inconsistent artifacts, as pointed out in [20, 24, 34]. Though Wavelet compression is able to provide better quality [13], it could still suffer from inconsistent artifacts that are difficult to measure. Distribution-based representations are better able to alleviate artifacts and are able to provide a confidence measure in visualization tasks. Thompson et al. [26] used histograms as a compact representation to represent samples in a local block from a deterministic dataset or in a voxel from an ensemble dataset. Liu et al. [20] modeled the samples in a local block or in a voxel from uncertainty datasets as a Gaussian Mixture Model. Dutta et al. [9] also partitioned the dataset into local blocks and modeled the data in each local block with the Gaussian Mixture Model in an in-situ simulation to save storage and I/O time. However, the common disadvantage of these approaches is the lack of spatial information in the region represented by a distribution, which leads to missing features, visual artifacts in renderings, and misleading users when used for analysis.

Distribution-based approaches for analysis: Distribution-based representations have also been used with other types of datasets and to assist in various analysis tasks. Chen et al. [8] improved pathline computation by interpolating from temporally down-sampled data and model the error with a Gaussian distribution. Chaudhuri et al. [6, 7] and Lee et al. [17] used a histogram based Summed Area Table to access the histogram of any sub-volume of the data set, which can be used to support a variety of data analysis tasks. Their work focused on the efficient access of local distributions rather than compact representation. Wei et al. present an efficient algorithm to search similar distributions of local regions [30]. Tikhonova et al. [27, 28] proposed a histogram representation for view-dependent attenuation to achieve fast transfer function exploration.

Distribution-based visualization: Thompson et al. [26] used a histogram representation to approximate topological structure and fuzzy isosurfaces. Liu et al. [20] used flickering in animations to convey the uncertainty in down-sampled or ensemble datasets. Lundström et al. [21] presented a technique using probabilistic animations to visualize uncertainty in medical volume datasets. Athawale et al. [1, 2] used statistical isosurface visualization techniques that compute the probability of an iso-value crossing at each voxel and visualize the probability field. He et al. [14] use particle filtering to visualize streamlines from datasets with uncertainty. They utilized distribution-based querying to find the regions with certain types of distributions [15]. Lehmann and Theisel [18] proposed a framework which explores and visualizes multivariate data sets from enhanced continuous parallel coordinates.

3 Overview

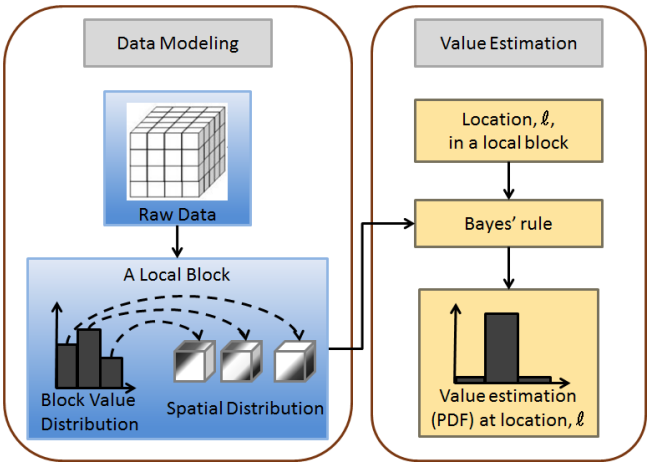


Figure 2: Overview of our approach. In a pre-processing step (shown on the left), the dataset is subdivided into blocks and both a value distribution and spatial distributions (Spatial GMMs) are computed for each block. To estimate a value at a given location, Bayes' rule is used to integrate the value and spatial distributions to obtain the probability density function of values at this location (shown on the right).

Our representation combines two kinds of distributions, the value distribution and spatial distributions. Figure 2 illustrates our representation and approach where our augmented representation is used to approximate a data value at a given location. The representation is constructed in a pre-processing step where the dataset is first subdivided into blocks. For each block, a value distribution is computed from the values of the samples in the block. This is a histogram consisting of B bins, where each bin represents a data value range as an interval. If b_i denotes the i^{th} bin, then the bin interval for b_i represents a data value range $[Lower_{b_i}, Upper_{b_i}]$. If M denotes the number of grid sample points in the block that have a value within the interval $[Lower_{b_i}, Upper_{b_i}]$ and N denotes the total number of

grid sample points of the block, then the i^{th} bin has a probability of M/N. In addition, at each bin a distribution of spatial locations is computed from the grid sample locations in the block, which provides the probability of the locations for the data values in the bin interval. Each spatial distribution modeled with a GMM, called a Spatial GMM, whose dimensionality is determined by the spatial dimensions of the data. For example, a 3D dataset will define Spatial GMMs with three dimensions. Details of the spatial distribution are introduced in Section 4. To determine a probability density function (PDF) to use for value estimation (see Section 5), we use Bayes' rule by integrating information from the value distribution and the Spatial GMMs.

4 Spatial Distribution

In this section, we describe the Spatial GMM associated with each bin in the value distribution and show how to use the Gaussian Mixture Model to compactly represent the distribution. The number of Gaussian components to use for a Spatial GMM should vary from one Spatial GMM to another in order to accurately represent coherence of the data values across the block represented by the bin interval. Fewer components should be used when data values are distributed coherently in space. Figure 3 illustrates the process of computing the spatial GMM on a local data block. Details of the data structure are presented at the end of this section.

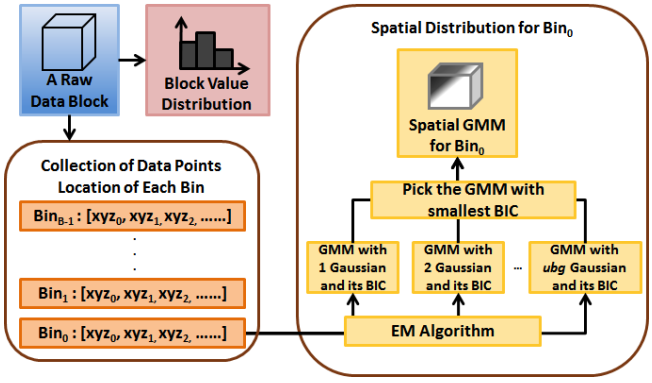


Figure 3: This diagram shows the steps used to compute the spatial GMM for a raw data block (shown in blue). Besides the computation of the value distribution, the raw data in the block is used to construct the Spatial GMM. First, the locations of the data samples are collected into the corresponding bin interval according to the data value at that location (shown in the bottom left). Then, a Spatial GMM is constructed (shown on the right) for each bin interval using the locations in the interval (illustrated here for Bin_0).

4.1 Spatial Distribution Per Bin Interval

The spatial distribution associated with a bin describes the probabilities of the locations for the values in the bin interval. The spatial distribution of a particular bin b_i can be modeled from the coordinates of the sample grid points whose scalar values belong to interval represented by bin b_i by using the Kernel Density Estimation (KDE) [12]:

$$S_{b_i}(x) = \frac{1}{N}\sum_{j=1}^{N} K_h(x - x_j) \tag{1}$$

where x is an arbitrary location in the block. Equation 1 computes S_{b_i}, which is the spatial distribution of bin b_i. $S_{b_i}(x)$ describes how likely x has a value within the interval b_i. Value N is the total number of grid points that have values within the bin interval b_i, x_j is the j^{th} grid point, and K_h is a kernel function.

4.2 Spatial Gaussian Mixture Model

A representation constructed directly from Equation 1 would incur a high storage cost to store information for N grid points. Instead, we can use the multivariate Spatial Gaussian Mixture Model [4], which compactly captures spatial coherence amongst similar data values, such as those in the same bin interval. The GMM is a widely used probabilistic model [9, 20, 31] because it can fit complex data well with a few number of parameters.

As opposed to the block GMM [9,20], which maps a data value in the block to a probability, the Spatial GMM maps a spatial location to a probability. The Spatial GMM is a multivariate GMM where the dimensionality is determined by the spatial dimensions of the dataset. A Spatial GMM for a location x is expressed in Equation 2

$$SGmm_{b_i}(x) = \sum_{j=1}^{K} w_j * \mathcal{N}(x|\mu_j, \Sigma_j) \tag{2}$$

where $SGmm_{b_i}(x)$ is the Spatial GMM of bin b_i, K is the number of Gaussian components, and w_j, μ_j and Σ_j are the weight, mean and covariance matrix for the j^{th} Gaussian component. $\mathcal{N}(x|\mu_j, \Sigma_j)$ is the probability density at x of the Gaussian distribution, which is defined by μ_j and Σ_j. For a three dimensional dataset, μ_j are the location coordinates and Σ_j is a 3x3 covariance matrix. $\sum_{j=1}^{K} w_j$ has to be equal to 1.

To obtain the weights, means and covariance matrices, of the Spatial GMM, we use the Expectation-Maximization (EM) algorithm [4] where the locations of the grid points whose values are within the interval of bin b_i are collected as the input training samples to find the parameters of $SGMM_{b_i}$. The EM algorithm iteratively maximizes the likelihood function described in Equation 3

$$\text{argmax}_\theta SGmm_\theta(X) = \sum_{m=1}^{M} \log[\sum_{k=1}^{K} w_k \mathcal{N}(x_m|\mu_k, \Sigma_k)] \tag{3}$$

where θ represents the parameters of the Spatial GMM. The EM algorithm iteratively adjusts and finds the parameters of the Spatial GMM by maximizing the likelihood of input samples $X = \{x_1, x_2, ..., x_M\}$. M is the number of grid points whose values are within the bin interval and x_m is the location coordinate of these input samples. By maximizing Equation 3, the difference between S_{b_i} and $SGmm_{b_i}$ is minimized.

The EM algorithm may suffer from large computational overhead due to the number of iterations needed for the algorithm to converge. Though, the convergence rate of the algorithm depends on the initial parameters [16, 33], where good initial parameters result in fewer iterations needed for convergence. In general, finding good initial parameters is difficult. However in our case, we observe that the parameters $SGmm_{b_i}$ can be a good initial guess for parameters $SGmm_{b_{i+1}}$ because the locations of the samples belonging to neighboring bins b_i and b_{i+1} are usually close to each other in scientific datasets. We can leverage this property to help the EM algorithm to quickly estimate the parameters and save on training time.

4.3 Varying Number of Gaussian Components

An advantage of using the Gaussian Mixture Model is the ability to approximate any arbitrary target distribution. In general, using more Gaussian components can fit the target distribution better. Using too many Gaussian components may not significantly improve the approximation quality, but it will increase storage overhead costs and sometimes causes over-fitting.

In order to decide on a suitable number of Gaussian components to be used in a Spatial GMM, we use Bayesian Information Criterion (BIC) [22] to choose the number of Gaussian components. Given a model and data, i.e. a Spatial GMM and the input training samples,

BIC will reward with a high likelihood over training samples and penalizes with a higher number of Gaussian components. In our implementation, an upper bound on the number of Gaussian components, called *ubg*, is given manually before training the Spatial GMMs. We iteratively check the BIC scores by changing the number of Gaussian components used for training a Spatial GMM. The number of Gaussian components with the smallest BIC is then selected for that Spatial GMM because lower BIC scores mean a better result for fitting. Hence, each Spatial GMM can have a different number of Gaussian components, which is determined by the complexity of the input data. This scheme avoids using too many Gaussian components without gaining significant quality improvement compared to using a fixed number of Gaussian components.

4.4 Data Structure

The data structure for the Spatial GMM is illustrated in Figure 4. Each Gaussian component stores a weight, a mean (location), and a covariance matrix. Because the covariance matrix is symmetric, only the upper triangle of the matrix is stored. Each bin consists of the adjusted probability of the histogram, the Gaussian component count of the GMM, and the Gaussian components.

Note that we store the adjusted probability instead of the original probability of the histogram in our data structure because this can preserve the correctness of the PDF computation, as described in the next section. Ideally, we would like the Spatial GMM to only model the space in a block region, but the Spatial GMM model is an approximate distribution representation and the domain is defined in infinite space. The Spatial GMM could give probabilities to coordinates outside the block and introduce bias in the PDF computation. Hence, we store the adjusted probability, $H'(b_i)$, to correct this, as shown in Equation 4

$$H'(b_i) = \frac{H(b_i)}{\int_\Omega SGmm_{b_i}(l)dl'} \tag{4}$$

where $H(b_i)$ is the original probability in the histogram, Ω is the spatial region of the block and $\int_\Omega SGmm_{b_i}(l)dl'$ is the accumulated probability over Ω of $SGmm_{b_i}$. In Section 5.1, we will show how to compute the PDF using $H'(b_i)$ and $SGmm_{b_i}$ in order to avoid the bias.

Different bins may have differing numbers of Gaussian components (see Section 4.3). So, a bin with zero probability requires storing zero Gaussian components. A homogeneous block, in which only one bin has probability, will not need the Spatial GMM. Sine the number of Gaussian components may differ per bin, the data size of each block varies as well, which leads to an irregular storage footprint. To easily access the data within blocks, an additional index table is stored where each index points to the starting location of the data in a block.

5 Probability of Data Values From Block and Spatial Distributions

This section describes how to to estimate a value at given a location within a block by combining information in the value distribution and the Spatial GMMs. Using Bayes' rule, we compute a probability density function (PDF) that provides probabilities for each possible value.

5.1 PDF Computation by Bayes' Rule

Bayes' rule is a popular theorem that is widely used in classification problems. It tells us how to rationally augment the known information with additional evidences from a given condition. In our scenario, the block value distribution is the known information and the additional evidences are the probabilities from each Spatial GMM at a given location. Bayes' rule is written as the random variable form

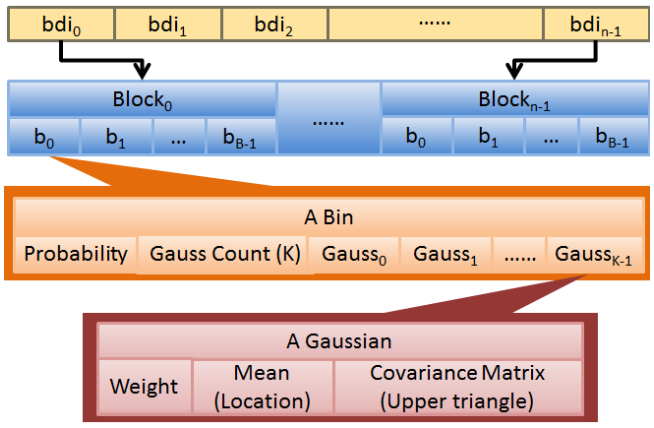


Figure 4: Data structure for our data representation using Spatial GMMs. Each block is indexed with a table of the starting locations of each block (shown at the top). Each block (shown in blue) stores bins (bin b_0 is shown in orange for $Block_0$). Each bin stores K Gaussians of the GMM for the bin. The data structure of a Gaussian component is shown in red.

$$P(I = b_i | L = \ell) = \frac{f_L(\ell | I = b_i) P(I = b_i)}{f_L(\ell)} \tag{5}$$

$$f_L(\ell) = \sum_{k=0}^{B-1} f_L(\ell | I = b_k) P(I = b_k) \tag{6}$$

where I is a random variable that represents a bin index, b_i is the i^{th} bin index, L is a random variable that represents a location in the spatial space of the block, and ℓ is the location in question. Given a data value within the interval of bin b_i, $f_L(\ell | I = b_i)$ represents the probability that the value of b_i is at the location ℓ. $P(I = b_i)$ is the probability of bin b_i of the value distribution. $f_L(\ell)$ sums up the numerator term in Equation 5 over all bins b_i for normalizing $P(I = b_i | L = \ell)$.

The function $f_L(\ell | I = b_i)$ comes from the normalized Spatial GMMs, $\frac{SGmm_{b_i}(\ell)}{\int_\Omega SGmm_{b_i}(l)dl'}$, which ignores the probability outside the block region. $P(I = b_i)$ comes from the value distribution. Equation 5 can be rewritten as Equation 7

$$\begin{aligned} P_\ell(b_i) &= \frac{\frac{SGmm_{b_i}(\ell)}{\int_\Omega SGmm_{b_i}(l)dl'} * H(b_i)}{\sum_{k=0}^{B-1} \frac{SGmm_{b_i}(\ell)}{\int_\Omega SGmm_{b_i}(l)dl'} * H(b_k)} \\ &= \frac{SGmm_{b_i}(\ell) * H'(b_i)}{\sum_{k=0}^{B-1} SGmm_{b_k}(\ell) * H'(b_k)} \end{aligned} \tag{7}$$

where $P_\ell(b_i)$ is the probability that the data sample at location ℓ has a value that belongs to the value sub-range of bin b_i, $H(b_i)$ is the probability associated with bin b_i in the value distribution, and P_ℓ is a PDF, which can be used to estimate the probabilities of various data values at location ℓ. The denominator normalizes $\sum_{k=0}^{B-1} P_\ell(b_k)$ to 1. $H'(b_i)$ is the adjusted probability of the histogram, which is mentioned in Section 4.4. The flow chart shown in Figure 5 illustrates how to compute the PDF at a given location ℓ.

6 Quantitative Evaluation

We show results from our experiments utilizing four datasets to analyze the storage cost of our representation and the quality of the

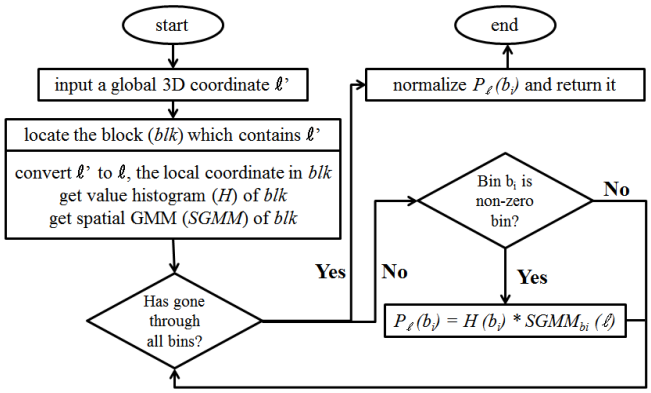


Figure 5: This flow chart shows the steps to compute the PDF consisting of probabilities associated with possible data values at a given a location ℓ. The input of this algorithm is a 3D location ℓ.

value estimation. We used an Intel 8-Core i-7-4770 3.40GHz CPU with 16 GBs of main memory and an NVIDIA GeForce GTX 660 video card with 2 GBs of video memory in our experiments. The Plume dataset was provided by the National Center for Atmospheric Research and is a simulation of Solar Plume for thermal down flow on the surface layer of the Sun. The data resolution is 504x504x2048 (the raw size is 1984 MB) and where we used the vector magnitude field in our experiments. The Isabel dataset is a pressure field of Hurricane Isabel from the IEEE Visualization 2004 Contest with a resolution of 500x500x100 (the raw size is 95 MB). The Combustion dataset was provided by Sandia National Laboratories and the mixture fraction field was used. The dataset has resolutions of 480x720x120 (the raw size is 158 MB). The Turbine dataset is a turbine engine compressor simulation which was used in [9]. The original Turbine dataset is a curvilinear grid data, and its Pressure variable is re-sampled to a regular grid. The resolution of the regular grid Turbine dataset is 2545x2545x440 (the raw size is 10871MB).

6.1 Representation Quality

We illustrate the improved accuracy of our representation using the Spatial GMM over two popular distribution-based representations: the block histogram, also called Hixel in [26], and the block GMM [9, 20]. We also compare our representation with histogram based trilinear interpolation (described in [23]) applied to the block histogram. Since a fundamental flaw of the block histogram is loss of spatial information, interpolation may compensate for this to some extent.

To evaluate the quality of data value estimation at given locations, we use the Root Mean Square Error (RMSE) metric, which encodes both bias and variance [29] as shown in equation 8

$$RMSE(P_\ell, x_\ell) = \sqrt{Bias(P_\ell, x_\ell)^2 + Variance(P_\ell)} \tag{8}$$

where the variable ℓ is a given location, P_ℓ is a PDF that indicates the possible values at location ℓ and their associated occurrence probability, and x_ℓ is the true value from the raw data. P_ℓ with a lower RMSE indicates that the estimate of a data value is closer to the true value and has smaller uncertainty.

We use Equation 7 to compute the PDF P_ℓ using our representation. The block histogram and block GMM representations directly use the corresponding block distribution of location ℓ as the basis to estimate the data value at location ℓ. P_ℓ is the interpolated histogram at ℓ when we used the block histogram with interpolation approach. The RMSE is calculated directly from the PDF at a location ℓ and the true value, as shown in equation 9.

$$RMSE(P_\ell, x_\ell) = \sqrt{\int_{-\infty}^{\infty} P_\ell(v) * (v - x_\ell)^2 dv} \tag{9}$$

Figure 6 shows the improved estimation accuracy of using our Spatial GMM representation (bottom three curves) when compared with the the block histogram and block GMM distribution-based representations (top three curves). Each curve has four points that represent the block sizes used in the experiment. Each point is a plot of the storage cost vs. RMSE using a particular block size. The RMSE in Figure 6 is the average of the RMSE values at all grid points in the dataset. The histograms for all representations used 128 bins and the block GMM used 5 Gaussian components (as was used in [9]). The curves representing the block histogram with and without interpolation have the same storage cost since the same block histograms were used in both cases. Given the same block size, block GMMs use less storage because the block histogram is a more compact representation. Eventhough our representation requires additional storage cost to incorporate spatial information, it significantly improves the RMSE. The bottom three curves show the estimation accuracy and storage cost of our Spatial GMM using up to 1, 3, and 5 Gaussian components for the spatial distributions. The spatial GMMs achieve lower RMSE when compared with all the other representations using just 1 Gaussian component, as illustrated by the red curve. In addition, given the same storage cost as the other approaches, our representation has smaller RMSE with different numbers of Gaussian components set as the upper bound. This trend can be observed consistently in all test data sets, which indicates that our approach gives better trade off between the quality and storage cost.

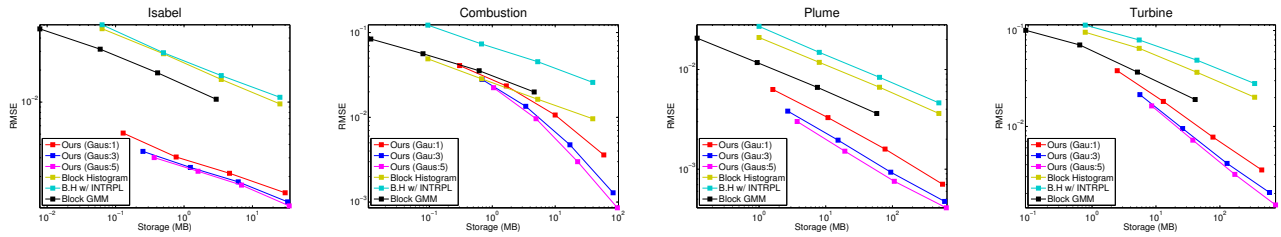
We also performed experiments to study the benefit of using the spatial distribution for different block sizes. We compared the RMSE of using only a block histogram compared with using a block histogram with our Spatial GMM, as shown in Figure 7. It is clear that by combining our Spatial GMM with the block histogram, we achieve a smaller RMSE when compared to only using a block histogram. Another observation is that when the block size is larger, the difference in RMSE with without our Spatial GMM is larger. The reason being that when a larger block size is used, more spatial information is lost, and thus we can benefit more from the spatial information found in the Spatial GMM.

6.2 Varying vs. Fixed Number of Gaussian Components

In this experiment, we show the benefit of varying the number of Gaussian components for the different spatial GMMs (see Section 4.3) rather than using the same or fixed number of Gaussian components for all of the Spatial GMMs. Using an upper bound on the number of Gaussian components (ubg) of 4 and compared with the fixed number of Gaussian components scheme, varying number of Gaussian components saves 29%, 28%, 16% and 13% Gaussian components on the Plume, Isabel, Combustion and Turbine datasets, respectively. We also compare these two schemes by varying different parameters such as block size and the upper bound on the number of Gaussian components. Both alternatives are compared with the $RDperMB(g)$ metric shown in Equation 10

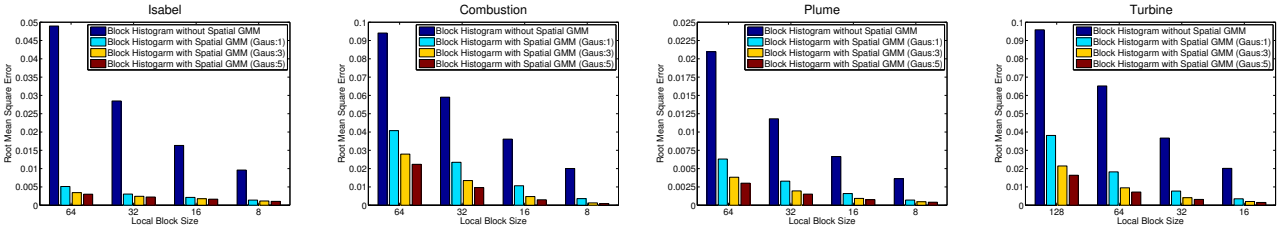
$$RDperMB(g) = \frac{RMSE(0) - RMSE(g)}{Size(g) - Size(0)} \tag{10}$$

where $RDperMB(g)$ stands for "RMSE Decrease per MBytes" and g represents the Gaussian components of a Spatial GMM, which is a fixed value when using a fixed number of Gaussian components and an upper bound when using a varying number of Gaussian components. $Size(g)$ is the storage cost using the Spatial GMM. The value $g = 0$ corresponds to using the block histogram only, i.e. no spatial information. Note that a higher $RDperMB(g)$ value



(a) Isabel RMSE (b) Combustion RMSE (c) Plume RMSE (d) Turbine RMSE

Figure 6: The trade off between accuracy (RMSE) and storage cost when comparing our spatial GMM approach using different block sizes and current approaches. Block sizes from left to right for (a),(b) and (c) are 64^3, 32^3, 16^3 and 8^3. Block sizes from left to right for (d) are 128^3, 64^3, 32^3 and 16^3. The bottom three curves show this trade off using our approach with upper bounds of 1, 3 and 5 on the number of Gaussian components.



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

Figure 7: Comparison of RMSE when block histogram with our spatial GMM and without our spatial GMM are used under different block sizes.

corresponds to better storage utilization in order to further decrease RMSE.

Figure 8 shows the decrease in RMSE per MBytes using both schemes. Four pairs of curves are shown for each dataset where the solid curves represent results from using a varying number of Gaussian components and the dotted curves represents the fixed number scheme. Each pair of curves, from the top to the bottom, was computed using the same block size. Varying the number of Gaussian components provides the same or larger decrease in $RDperMB(g)$, as shown by the the solid curves always being above the dotted curves. This is observed for all block sizes and is more pronounced for smaller block sizes. In addition, this is also observed when using a larger upper bound on the number of Gaussian components, indicated by the horizontal axis in the plots. In many cases, the spatial information in a smaller local block is more easily modeled so that using more Gaussian components may not be a good trade off. Varying the number of Gaussian components allows for fewer components and the use of more components on a need only basis.

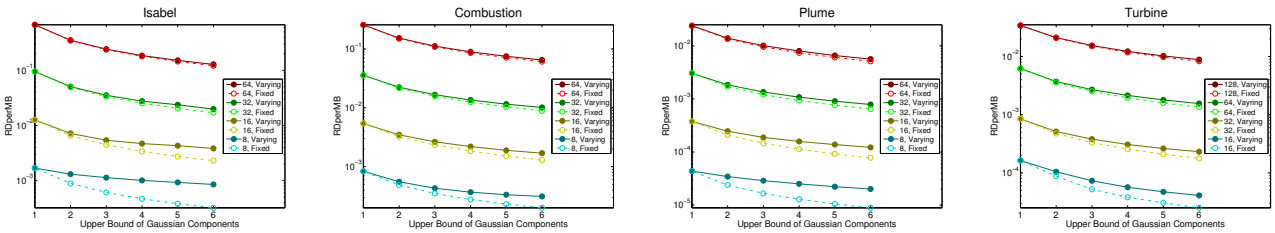
7 Visualization and Analysis

This section compares our approach to the block histogram and block GMM approaches [9, 20] on three data visualization and analysis applications. The first application is volume rendering of a reconstructed scalar field generated by sampling values from a distribution based representation. The second application is uncertain isosurface generation by computing the level crossing probability [2] at each voxel. The third application is local distribution-based feature matching. For each of these three applications, we first construct the appropriate field from a dataset using user provided parameters for the visualization task and a user defined region to construct the field and its resolution. Thus, scalar, probability, and similarity fields are constructed for the first, second, and third applications, respectively. Once constructed, we render the field with Paraview and perform viewpoint and transfer function exploration. We use the entire region of the dataset to construct each field. We use the same datasets used in our experiments in Section 6. We used the same spatial resolution as the original raw datasets to construct the fields for the Combustion, Isabel and Plume datasets. The Turbine dataset uses 1272x1272x220 as the spatial resolution. In this experiment, we set the local block size to 16^3 for the Combustion, Isabel and Plume datasets and 32^3 for Turbine dataset for our approach. We set the number of bins for the histograms to 128 and the upper bound of the number of Gaussian components for each Spatial GMM to 4. The datasets Plume, Isabel, Combustion and Turbines have averages of 51.03, 16.04, 10.36 and 31.38 Gaussian components per block, respectively. For the block histogram approaches, the histogram has 128 bins. We used 5 Gaussian components for each GMM in the block GMM approach. We ensured that all three representations, the block histograms with interpolation, the block histogram without interpolation, and block GMMs all had close storage size as our representation by adjusting the local block size.

7.1 Volume Rendering

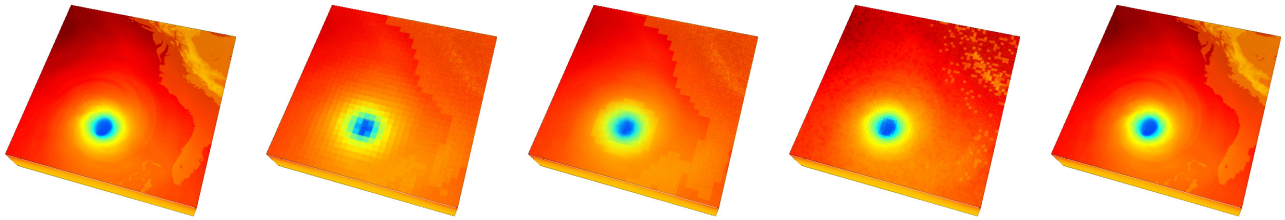
Our first experiment is to use each of the distribution-based representations for a direct volume render volume rendering task. We reconstruct the scalar field by drawing samples from the computed PDF of our representation, and from the histogram, interpolated histogram and GMM of other these approaches.

Figures 9 - 12 show renderings of each dataset (Isabel (Figure 9), Combustion (Figure 10), Plume (Figure 11) and Turbine (Figure 12)) using the four approaches. Renderings computed with the block histogram without interpolations (shown in (b)) clearly shows blocky artifacts and lost detail. Renderings using the block histogram with interpolations (shown in (c)) alleviate the blocky artifacts, but details are blurred. Renderings computed with the block GMM (shown in (d)) show clear improvement from the block histogram only approaches. However, the approach also suffers from blocky artifacts and noise. Our approach (shown in (e)) is a marked improvement over the other approaches. Though our results do not exactly match the ground truth renderings (shown in (a)), it is much closer with a



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

Figure 8: Comparison of varying and fixed number of Gaussian components schemes. The curves with same color use the same local block size. The solid and dotted lines indicate the representations generated from the varying and fixed number of Gaussian components schemes respectively.



(a) Rendering from raw data (b) Block histogram (7.93MB) (c) Block Hisotgram w/ interpolation (7.93MB) (d) Block GMM: (7.03MB) (e) Our approach (6.76MB)

Figure 9: Visual comparison of volume rendering in Pressure variable of Isabel dataset. The samples are drawn from the PDFs, which are calculated at all grid points of the raw data, using Monte Carlo sampling. The block size of (b),(c),(d) and (e) are 12^3, 12^3, 6^3 and 16^3, respectively. In (e), the value histogram costs 3.5MB and the Spatial GMM costs 3.26MB.

little noise, as illustrated in the zoomed in views.

7.2 Uncertain Isosurface

Our next experiment is to show a visual comparison using uncertain isosurface rendering between the approaches using a statistical isosurface visualization technique with uncertain realizations [1, 2, 26]. We use the uncertain isosurface technique proposed by Athawale et al. [1] which computes the probabilities of the isovalue crossing in each cell and renders the probability field. We use Equation 11 to compute uncertain isosurface

$$P_{crossing}(c) = 1 - \prod_{k=0}^{7} \int_{-\infty}^{c} P_{\ell_k}(v)dv - \prod_{k=0}^{7} \int_{c}^{\infty} P_{\ell_k}(v)dv \qquad (11)$$

where c is an isovalue and $P_{\ell_0}...P_{\ell_7}$ are the PDFs for the data values at the eight grid points on the cell.

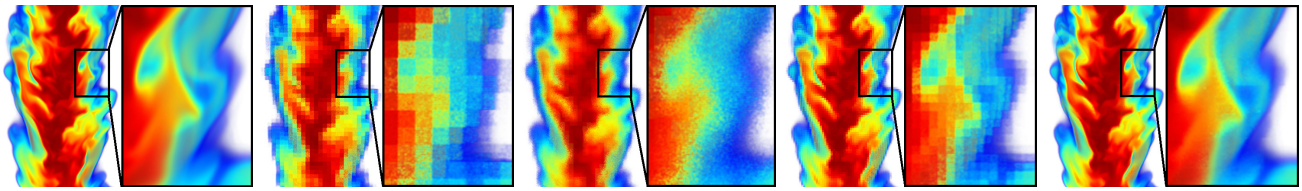
Figure 1 shows renderings using the four approaches on the Isabel dataset. Figure 1 (a) and (f) show isosurface renderings generated from the original raw data to represent the ground truth. As was seen in the volume rendering results, renderings computed using the block histogram (Figure 1 (b) and (g)) and the block GMM (shown in Figure 1 (d) and (i)) approaches clearly suffer from blocky artifacts. Interpolation applied to the block histogram (Figure 1 (c) and (h)) greatly alleviates these blocky patterns, but the result only shows a rough uncertain isosurface. Our approach is able to show the isosurfaces much more clearly and accurately (Figure 1 (e) and (j)). Rendering using the Combustion dataset are shown in Figure 13. Notice the two holes in the zoomed in region for the ground truth rendering in (a). This feature is completely or nearly absent in the renderings using the other approaches (Figure 13 (b) (c) and (d)) due to higher uncertainty from these representations. On the contrary, this feature is preserved using our approach (Figure 13) (e)).

7.3 Local Distribution-based Feature Matching

One application of local distribution-based features [7, 15, 30] is to identify locations where local histograms are similar to a target feature defined by the user [7, 30]. Given a target distribution and a neighborhood size, the local distribution at each voxel in the raw data domain is computed from its neighborhood, and the L1-norm distance measure is applied to compute the similarity between the local distribution and the target distribution. Figure 14 shows renderings of the similarity fields from the search results on the Combustion dataset. The neighborhood size is set to 5x5x5 and the target distribution is selected from a region with pure fuel mass in the mixture fraction variable of Combustion data. Figure 14 (a) shows a ground truth rendering of the search result applied to the raw data. Results when using the block histogram (figures 14 (b) and (c)) and the block GMM (Figure 14 (d)) are unsatisfactory due to areas that are overestimated in the region around the pure fuel mass as a result of using an improper PDF with large uncertainty to estimate a value at each voxel. As the zoomed in results show, our approach (Figure 14 (e)) is extremely close to the ground truth.

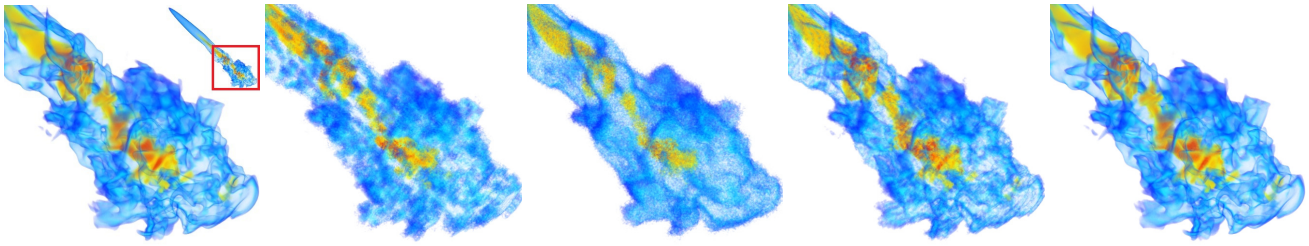
8 Discussion

Preprocessing Time: We discuss and report the preprocessing time per block of our approach for the four datasets used in our experiments. We measure the preprocessing time to compute our representation with the same settings reported in Section 7 and computed on the same machine reported in Section 6. In the preprocessing step, we subdivide the data into blocks, compute the block histogram and Spatial GMMs using our adaptive scheme described in Section 4.3. The program is written in *Python* 3.3 using the *mixture* package in the *scikit-learn* library [5] to train the Spatial GMM. Since preprocessing time is affected by data complexity, more complex data requires more time to run the EM algorithm in training the Spatial GMMs due to more iterations needed for the algorithm to converge. For example, the our algorithm took the most time to pre-process



(a) Rendering from raw data (b) Block histogram (20.73MB) (c) Block Histogram w/ interpolation (20.73MB) (d) Block GMM: (19.34MB) (e) Our approach (20.63MB)

Figure 10: Visual comparison of volume rendering in combustion dataset. The samples are drawn from the PDFs, which are calculated at all grid points of the raw data, using Monte Carlo sampling. The block size of (b),(c),(d) and (e) are 10^3, 10^3, 5^3 and 16^3, respectively. In (e), the value histogram costs 5.27MB and the Spatial GMM costs 15.36MB.



(a) Rendering from raw data (b) Block histogram (120.16MB) (c) Block histogram w/ interpolation (120.16MB) (d) Block GMM (141.39MB) (e) Our approach (103.17MB)

Figure 11: Visual comparison of volume rendering in Plume dataset. The thumbnail at right upper of (a) is the global view of Plume. We zoom in to the red box region in this visual comparison. We zoom in and render part of the dataset. The samples are drawn from the PDFs, which are calculated at all grid points of the raw data, using Monte Carlo sampling. The block size of (b),(c),(d) and (e) are 13^3, 13^3, 6^3 and 16^3, respectively. In (e), the value histogram costs 64MB and the Spatial GMM costs 39.17MB.

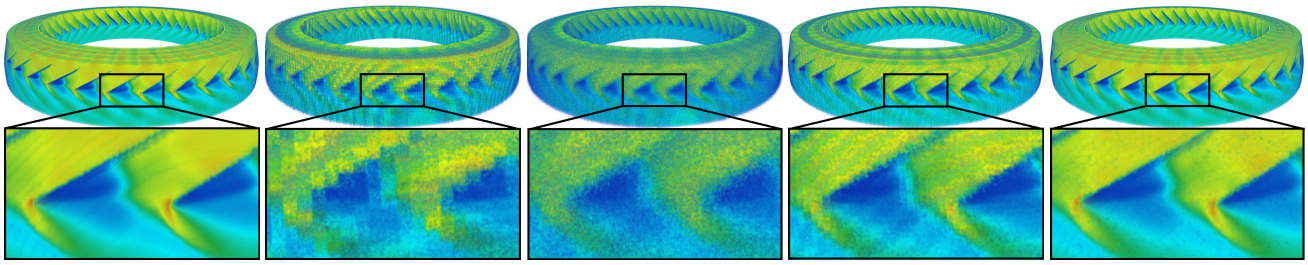
each block on average on the Combustion dataset since it has the most complexity amongst all of our test datasets. Computation time for each test dataset was 0.97 seconds to pre-process each block on average on the Combustion dataset, 0.51 second on the Plume dataset, 0.55 second on the Isabel dataset, and 0.69 second on the Turbine dataset.

PDF Computation Time: The time to compute the PDF depends on the number of Gaussian components. Hence, the time complexity of computing a PDF at a given location is $O(G)$, where G is the average number of Gaussian components per block and bin. We implemented the PDF computation using the NVIDIA CUDA Thrust Library and used the same machine as was used for the experiments conducted in Section 6 to compute run times here. We report both the total and average computation times on each dataset. Also, we use the same parameter settings that were used in Section 7. Since the resolutions of reconstructed fields are different, the total number of computed PDFs are also different. The total time to compute the PDF for the Combustion dataset took 3.79 seconds ($9.15 * 10^{-8}$ second in average), 1.12 seconds ($4.48 * 10^{-8}$ second in average) for the Isabel dataset, 20.99 seconds ($3.91 * 10^{-8}$ second in average) for the Plume dataset, and 291.08 seconds ($8.17 * 10^{-7}$ second in average) for the Turbine dataset. Note that the times reported here only include the time to compute the PDF computation.

We also report the total time to construct the scalar field in Section 7, which includes PDF computation time and value sampling time from the distribution. The time measurements for Combustion, Isabel, Plume and Turbine are 5.19, 1.89, 32.81 and 351.67 seconds, respectively. After the field is constructed and the two stage visualization pipeline (see Section 7) is used, both viewpoint and transfer function exploration can be computed in real-time.

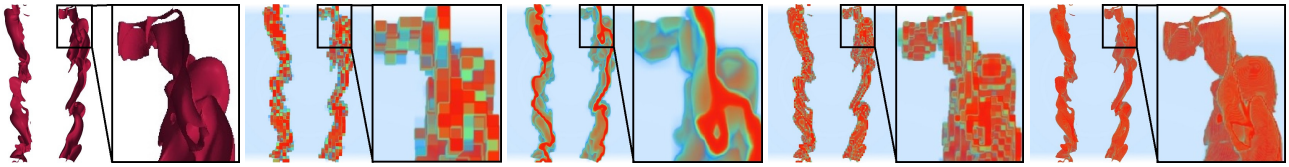
Influence of Parameters: Three important parameters are the number of bins, the upper bound of the Gaussian components, and block size. An obvious trade off is that by increasing the bin number, you will increase the size of the histogram and the number of Spatial GMMs needed, but improve precision in the value domain. The influence of the upper bound of Gaussian components of the Spatial GMM can be seen in our experiments. In Figure 7, the RMSE computed over the datasets is decreasing for all block sizes when the upper bound of Gaussian components of the Spatial GMM increases. The reason is that with more Gaussian components used in the Spatial GMM there can be a better approximation of the Spatial distribution. But the "RMSE Decreasing per MByte" keeps decreasing while the upper bound of Gaussian components increases, as shown in Figure 8. This means that the per unit storage cost for the Spatial GMM results in decreasing quality improvement when the upper bound of Gaussian components is increasing. As mentioned previously, the PDF computation time is proportional to the total number of Gaussian components. Thus, increasing the upper bound of Gaussian components helps to improve quality but is affected by diminishing returns and increases PDF computation time. Also, an infinitely small block size makes the spatial distribution useless because no spatial information is lost. In Figure 8, we observe that the curves showing larger block sizes are always on the top of the curves showing smaller block sizes. This means that the per unit storage cost for the Spatial GMM improves quality more. But, Figure 7 also shows that the smaller block size always has better RMSE. Hence, when a larger block size is used, we can gain more by using the Spatial GMMs, but the smaller block size can provide better overall RMSE.

Scalability: Since computation of the distributions is done in a separate pre-processing step and distributions are localized to individual blocks, our approach scales well and easily leverages parallel processing. Pre-processing tasks over a single block are independent of pre-processing tasks in other blocks and only requires the data within that local block. Thus, the memory requirement for pre-processing each block only depends on the local block size and



(a) Rendering from raw data (b) Block histogram (131.4MB) (c) Block histogram w/ interpolation (131.4MB) (d) Block GMM (163.71MB) (e) Our approach (151.54MB)

Figure 12: Visual comparison of volume rendering in Turbine dataset. The samples are drawn from the PDFs, which are calculated at all grid points of the raw data, using Monte Carlo sampling. The block size of (b),(c),(d) and (e) are 22^3, 22^3, 10^3 and 32^3, respectively. In (e), the value histogram costs 43.75MB and the Spatial GMM costs 107.79MB.



(a) True isosurface (b) Block histogram (20.73MB) (c) Block Hisotgram w/ interpolation (20.73MB) (d) Block GMM (19.34MB) (e) Our approach (20.63MB)

Figure 13: Uncertain isosurface (isovalue = 0.15) visualization of the Combustion dataset. (a) is the true isosurface from the raw data. The sizes of the distribution-based representations were similar in the results shown in (b),(c),(d), and (e). The color orange indicates locations with higher possibility for the location of the isosurface. The color blue indicates locations with less possibility for the isosurface. The settings used here are the same as used for the results in Figure 10.

does not increase with raw data size. The computation time for pre-processing is proportional to the number of data points within a local block and the number of blocks of the dataset. Although the time to process each Spatial GMM in a local block is difficult to estimate, as it depends on the parameters of EM algorithm and properties of the data, the computation time of a Spatial GMM increases in general with more data points in a local block, and vice versa. The pre-processing time over the entire dataset grows just linearly with the number of blocks. Computing the pre-processing is suitable on multi-node supercomputers to bound the overall pre-process time for large datasets. Since no communication is needed between processes, no specially designed parallel algorithm is needed to apply our algorithm for pre-processing to multi-node supercomputers.

9 Conclusion and Future Work

This paper presented a novel distribution-based representation for large-scale data sets, which allows high-quality statistical based analyses and visualization. Our proposed Spatial GMM representation compactly stores spatial information, which is missing in current distribution-based representations. In order to keep the storage overhead small, an adaptive scheme is used to determine the number of Gaussian components needed for each Spatial GMM. We qualitatively compared our representation with existing distribution representations. Our approach is able to compute the probability density function of values at any location, which represents the possible values and its associated occurrence probabilities, with less bias and variance at each voxel and provides superior visual results in the three visual analysis applications in our experiments.

In future work, we will explore different distribution-based fitting models since datasets that have very complex or non-smooth data value profiles may not be represented well with our Spatial GMM representation. We would like to develop better models for spatial information and explore ways to gain more storage savings. Our approach can be extended to other types of datasets, such as time varying data, where our approach can be applied to individual time steps and share common histograms or Spatial GMMs to save on storage. Multivariate, vector, and higher dimensional volume datasets are other possibilities to apply our approach on. For multivariate and vector volume datasets, we need to address schemes for more efficient techniques to store the value distribution. For higher dimensional datasets, our approach is applicable by using the Spatial GMM with more variables.

Acknowledgments

This work was supported in part by NSF grants IIS- 1250752, IIS-1065025, and US Department of Energy grants DE- SC0007444, DEDC0012495,program manager Lucy Nowell.

References

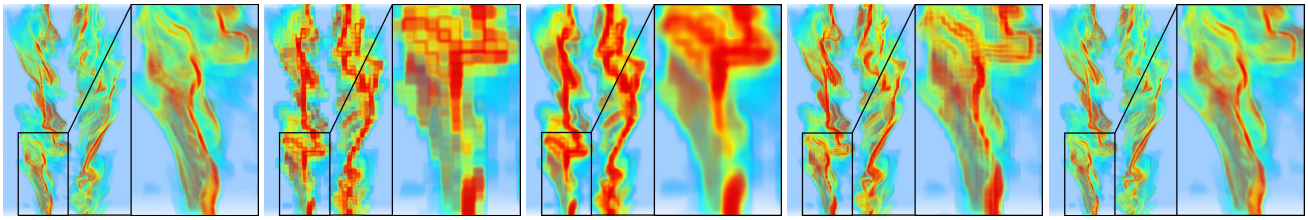
[1] T. Athawale and A. Entezari. Uncertainty quantification in linear interpolation for isosurface extraction. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2723–2732, 2013.

[2] T. Athawale, E. Sakhaee, and A. Entezari. Isosurface visualization of data with nonparametric models for uncertainty. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):777–786, 2016.

[3] A. C. Bauer, H. Abbasi, J. Ahrens, H. Childs, B. Geveci, S. Klasky, K. Moreland, P. O'Leary, V. Vishwanath, B. Whitlock, et al. In situ methods, infrastructures, and applications on high performance computing platforms. In *Computer Graphics Forum*, vol. 35, pp. 577–597. Wiley Online Library, 2016.

[4] J. A. Bilmes et al. A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden markov models. *International Computer Science Institute*, 4(510):126, 1998.

[5] L. Buitinck et al. Api design for machine learning software: experiences from the scikit-learn project. *arXiv preprint arXiv:1309.0238*, 2013.



(a) True Similarity Field (b) Block histogram (20.73MB) (c) Block Hisotgram w/ interpolation (20.73MB) (d) Block GMM (19.34MB) (e) Our approach (20.63MB)

Figure 14: Images of the distribution similarity field. (a) is the similarity field from the raw data. (b)(c)(d) and (e) are the similarity field from other distribution representation with similar storage size. Red color indicates high similarity. Blue color indicates low similarity. The data representations here are the setting in Figure 10.

[6] A. Chaudhuri, T.-Y. Lee, B. Zhou, C. Wang, T. Xu, H.-W. Shen, T. Peterka, and Y.-J. Chiang. Scalable computation of distributions from large scale data sets. In *Large Data Analysis and Visualization (LDAV), 2012 IEEE Symposium on*, pp. 113–120. IEEE, 2012.

[7] A. Chaudhuri, T. H. Wei, T. Y. Lee, H. W. Shen, and T. Peterka. Efficient range distribution query for visualizing scientific data. In *Visualization Symposium (PacificVis), 2014 IEEE Pacific*, pp. 201–208. IEEE, 2014.

[8] C.-M. Chen, A. Biswas, and H.-W. Shen. Uncertainty modeling and error reduction for pathline computation in time-varying flow fields. In *2015 IEEE Pacific Visualization Symposium (PacificVis)*, pp. 215–222. IEEE, 2015.

[9] S. Dutta, C. Chun-Ming, G. Hernlein, H.-W. Shen, and J. Chen. In situ distribution guided analysis and visualization of transonic jet engine simulations. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):837–846, 2016.

[10] N. Fout and K.-L. Ma. Transform coding for hardware-accelerated volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1600–1607, 2007.

[11] E. Gobbetti, J. A. Iglesias Guitián, and F. Marton. Covra: A compression-domain output-sensitive volume rendering architecture based on a sparse representation of voxel blocks. In *Computer Graphics Forum*, vol. 31, pp. 1315–1324. Wiley Online Library, 2012.

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

[13] S. Guthe and W. Strasser. Advanced techniques for high-quality multiresolution volume rendering. *Computers & Graphics*, 28(1):51–58, 2004.

[14] W. He, C.-M. Chen, X. Liu, and H.-W. Shen. A bayesian approach for probabilistic streamline computation in uncertain flows. In *2016 IEEE Pacific Visualization Symposium (PacificVis)*, pp. 214–218. IEEE, 2016.

[15] C. R. Johnson and J. Huang. Distribution-driven visualization of volume data. *IEEE Transactions on Visualization and Computer Graphics*, 15(5):734–746, 2009.

[16] D. Karlis and E. Xekalaki. Choosing initial values for the EM algorithm for finite mixtures. *Computational Statistics & Data Analysis*, 41(3):577–590, 2003.

[17] T.-Y. Lee and H.-W. Shen. Efficient local statistical analysis via integral histograms with discrete wavelet transform. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2693–2702, 2013.

[18] D. J. Lehmann and H. Theisel. Features in continuous parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):1912–1921, 2011.

[19] J. J. Lin, J. Pei, X. Hu, W. Chang, R. Nambiar, C. Aggarwal, N. Cercone, V. Honavar, J. Huan, B. Mobasher, and S. Pyne, eds. *2014 IEEE International Conference on Big Data, Big Data 2014, Washington, DC, USA, October 27-30, 2014*. IEEE, 2014.

[20] S. Liu, J. A. Levine, P. Bremer, and V. Pascucci. Gaussian mixture model based volume visualization. In *Large Data Analysis and Visualization (LDAV), 2012 IEEE Symposium on*, pp. 73–77. IEEE, 2012.

[21] C. Lundström, P. Ljung, A. Persson, and A. Ynnerman. Uncertainty visualization in medical volume rendering using probabilistic animation. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1648–1655, 2007.

[22] D. Posada and T. R. Buckley. Model selection and model averaging in phylogenetics: advantages of akaike information criterion and bayesian approaches over likelihood ratio tests. *Systematic biology*, 53(5):793–808, 2004.

[23] A. Read. Linear interpolation of histograms. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 425(1):357–360, 1999.

[24] R. Sicat, J. Kruger, T. Möller, and M. Hadwiger. Sparse pdf volumes for consistent multi-resolution volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2417–2426, 2014.

[25] K. Stockinger, J. Shalf, K. Wu, and E. W. Bethel. Query-driven visualization of large data sets. In *IEEE Visualization*, vol. 5, p. 22, 2005.

[26] D. Thompson, J. A. Levine, J. C. Bennett, P.-T. Bremer, A. Gyulassy, V. Pascucci, and P. P. Pébay. Analysis of large-scale scalar data using hixels. In *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*, pp. 23–30. IEEE, 2011.

[27] A. Tikhonova, C. Correa, and K.-L. Ma. Visualization by proxy: A novel framework for deferred interaction with volume data. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1551–1559, 2010.

[28] A. Tikhonova, C. D. Correa, and K.-L. Ma. An exploratory technique for coherent visualization of time-varying volume data. In *Computer Graphics Forum*, vol. 29, pp. 783–792. Wiley Online Library, 2010.

[29] D. Wackerly, W. Mendenhall, and R. L. Scheaffer. *Mathematical statistics with applications*. Nelson Education, 2007.

[30] T.-H. Wei, C.-M. Chen, and A. Biswas. Efficient local histogram searching via bitmap indexing. In *Computer Graphics Forum*, vol. 34, pp. 81–90. Wiley Online Library, 2015.

[31] B. Yegnanarayana and S. P. Kishore. Aann: an alternative to gmm for pattern recognition. *Neural Networks*, 15(3):459–469, 2002.

[32] H. Yela, I. Navazo, and P. Vazquez. S3dc: A 3dcbased volume compression algorithm. *Proc. CEIG*, 2, 2008.

[33] J. Yin, Y. Zhang, and L. Gao. Accelerating expectation-maximization algorithms with frequent updates. In *Cluster Computing (CLUSTER), 2012 IEEE International Conference on*, pp. 275–283. IEEE, 2012.

[34] H. Younesy, T. Möller, and H. Carr. Improving the quality of multiresolution volume rendering. In *EuroVis*, pp. 251–258. Citeseer, 2006.