Visual Analysis of Large Multivariate Scattered Data using Clustering and Probabilistic Summaries

Tobias Rapp, Christoph Peters, and Carsten Dachsbacher

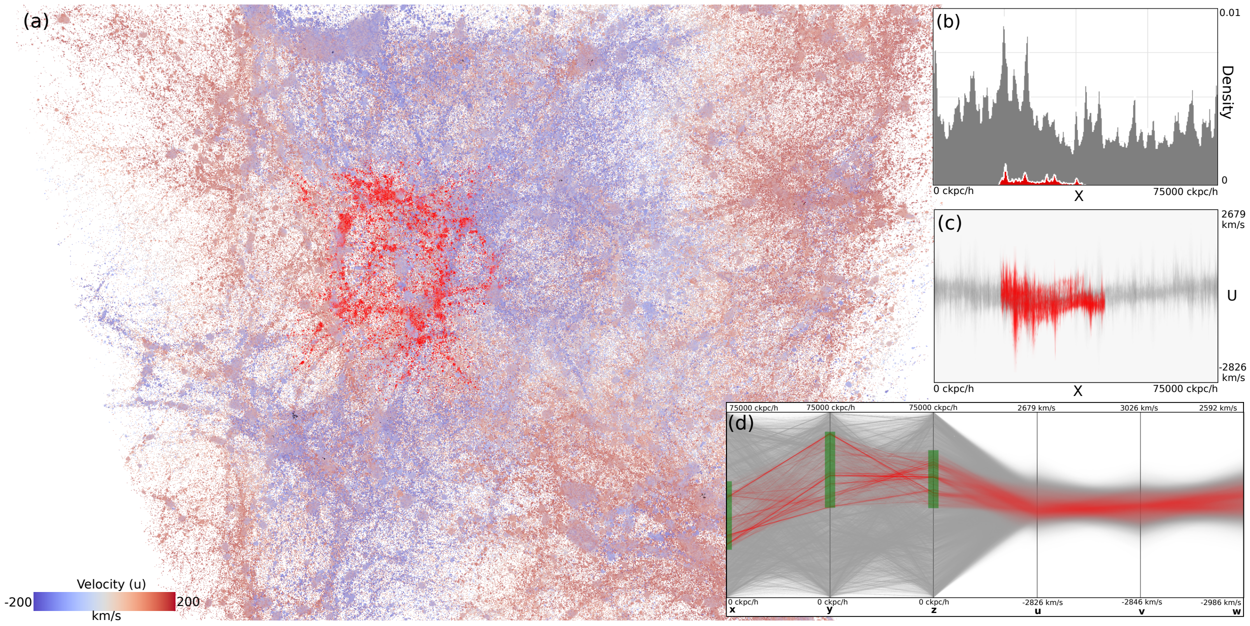


Fig. 1: Our probabilistic summary of a cosmological dataset represents 2.6 billion particles partitioned into 5.3 million clusters. We model each cluster using combinations of low-dimensional Gaussian mixture models. This allows us to interactively visualize the position of particles by splatting 3D Gaussians (a) and to create density-based 1D and 2D plots, depicted in (b) and (c). A density-based parallel coordinate plot is shown in (d). All of those views support interactive navigation and exploration by brushing (red) and linking. We render this massive dataset in 28 ms on an NVIDIA GTX 1080 Ti at a resolution of 1920×1080.

Abstract— Rapidly growing data sizes of scientific simulations pose significant challenges for interactive visualization and analysis techniques. In this work, we propose a compact probabilistic representation to interactively visualize large scattered datasets. In contrast to previous approaches that represent blocks of volumetric data using probability distributions, we model clusters of arbitrarily structured multivariate data. In detail, we discuss how to efficiently represent and store a high-dimensional distribution for each cluster. We observe that it suffices to consider low-dimensional marginal distributions for two or three data dimensions at a time to employ common visual analysis techniques. Based on this observation, we represent high-dimensional distributions by combinations of low-dimensional Gaussian mixture models. We discuss the application of common interactive visual analysis techniques to this representation. In particular, we investigate several frequency-based views, such as density plots in 1D and 2D, density-based parallel coordinates, and a time histogram. We visualize the uncertainty introduced by the representation, discuss a level-of-detail mechanism, and explicitly visualize outliers. Furthermore, we propose a spatial visualization by splatting anisotropic 3D Gaussians for which we derive a closed-form solution. Lastly, we describe the application of brushing and linking to this clustered representation. Our evaluation on several large, real-world datasets demonstrates the scaling of our approach.

Index Terms—interactive visual analysis, probabilistic data summaries, multivariate data, scattered data, Gaussian mixture models, Gaussian rendering

1 Introduction

The field of scientific visualization is confronted with rapidly growing amounts of data, including multivariate and time-dependent data. Interactive visual analysis [46] approaches have been established as a powerful approach to facilitate knowledge discovery in complex datasets. However, growing data sizes make the interactive exploration increasingly difficult or even impossible for some datasets.

To deal with large amounts of data, recent approaches employ probabilistic data summaries [7, 8, 12, 45] to represent blocks of data as probability distributions. These approaches have been mostly limited to univariate, volumetric data. In this work, we propose a representation that supports arbitrarily structured, time-dependent, and multivariate data defined in a two- or three-dimensional spatial domain. To this end, the data needs to be partitioned, i.e. clustered into spatially coherent regions. In each cluster, we make use of Gaussian mixture models

• Tobias Rapp, Christoph Peters, Carsten Dachsbacher are with Karlsruhe Institute of Technology. E-mail: tobias.rapp, christoph.peters, dachsbacher@kit.edu.

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

(GMMs) to compactly represent a probability distribution of the data using a weighted combination of Gaussian components. However, multivariate data requires modeling high-dimensional distributions, which suffer from the curse of dimensionality. Our approach is based on the observation that representations of low-dimensional marginal distributions suffice to analyze and visualize the data. All common visualizations, such as scatter plots, histograms, and parallel coordinate plots, require only 1D or 2D distributions. The exception is the spatial domain of scattered data in 3D for which we employ a 3D distribution. Thus, we model the marginal distributions of all individual dimensions and pairs of dimensions as well as the spatial 3D distribution.

For large data, common item-based visualizations, such as scatter and parallel coordinate plots, are challenged by overdraw and cluttering. Frequency-based visualizations are a viable alternative in this case [25,34]. Density estimation [40] is a frequency-based approach commonly used in statistics. However, its usage in interactive visualization has been limited due to performance considerations. Although our approach supports all common visualization techniques, it is especially well suited to density-based techniques since our modeled distributions are already an estimate of density. We discuss the efficient visualization and interaction with density-based plots using our compact representation. Additionally, we consider time-dependent histograms that would otherwise be infeasible to produce for large datasets. In this view, we can interactively brush over different time steps. To visualize the uncertainty introduced by our data representation, we propose an error metric based on the cumulative distribution function, similar to statistical goodness of fit tests. A level-of-detail mechanism allows scientists to drill down on interesting or uncertain regions in the data. Additionally, we discuss the explicit visualization of outliers, which are not handled well by density-based visualizations.

Our last contribution is the visualization of spatial density distributions. Since drawing and rendering samples from the GMMs would be infeasible for large, scattered datasets, we directly render 3D Gaussians. We derive a closed-form solution to integrate anisotropic Gaussians using a splatting approach. Back-to-front splatting has the disadvantage that it assumes non-overlapping Gaussians. Therefore, we employ moment-based order-independent transparency [31] for datasets where this is not an acceptable assumption.
To summarize, our main contributions are:

- We define compact data representations based on probabilistic models of low-dimensional marginal distributions for scattered, multivariate data,
- We describe interactive visual analysis techniques based on our probabilistic data summaries,
- We efficiently visualize scattered, overlapping, anisotropic 3D Gaussians.

2 Related Work

In this section, we discuss previous work on probabilistic data modeling, density-based visualizations, and the rendering of 3D Gaussians.

2.1 Probabilistic Modeling of Large Data

Several probabilistic approaches to represent large volumetric, univariate datasets have been proposed. Thompson et al. [42] describe *hixels*, a data representation that stores a histogram per block of voxels. Liu et al. [27] discuss volume rendering using per voxel Gaussian mixture models. Sicat et al. [39] construct a multi-resolution volume from sparse probability density functions defined in the 4D domain comprised of the spatial and data range. To visualize and analyze large volumetric data, Wang et al. [45] employ a spatial GMM in addition to a value distribution in each data block. For in-situ feature analysis of time-varying data, Dutta et al. [7] perform incremental GMM estimation instead of expectation maximization, which is traditionally used to estimate the parameters of a mixture model. By design, none of these approaches is applicable to more than four dimensions. Dutta et al. [8] model a single Gaussian or a GMM with a fixed number of components to each univariate value distribution in a cluster of the data. The authors compare several clustering techniques to determine homogeneous regions in volumetric data. Since an optimal clustering of the data is generally domain or application specific, we do not make any assumptions about the clustering procedure. Our method overcomes the limitation to low-dimensional data by working with low-dimensional GMMs for all relevant combinations of dimensions. We also introduce a fast and adaptive selection of the number of GMM components.

For parameter studies in cosmological simulations, Wang et al. [44] store GMMs as a prior knowledge to reconstruct high-resolution datasets from multiple prior simulation runs. Li et al. [26] reduce cosmological simulation data in-situ by subdividing space using a k-d tree and estimating particle density using a GMM in each leaf node. During the analysis stage, particles are sampled from the GMMs. Hazarika et al. [11] propose a copula-based uncertainty modeling approach to represent a multivariate distribution using different types of univariate distributions, including GMMs, separately from their interrelation. To summarize large-scale multivariate volumetric data, a copula-based analysis framework has been introduced [12]. This approach is the first to address the modeling of multivariate data, but the Gaussian copula function limits the correlations between dimensions to a single Gaussian. Whilst we similarly decompose a high-dimensional model into more manageable low-dimensional models, we do not share this limitation. Moreover, the approaches of Hazarika et al. and Li et al. require sampling, which hinders the application to interactive visual analysis, especially for rendering scattered data. Similarly, we do not perform subsampling of scattered data [36,48] for data reduction since our GMMs already estimate density, which we use directly in our density-based visualizations.

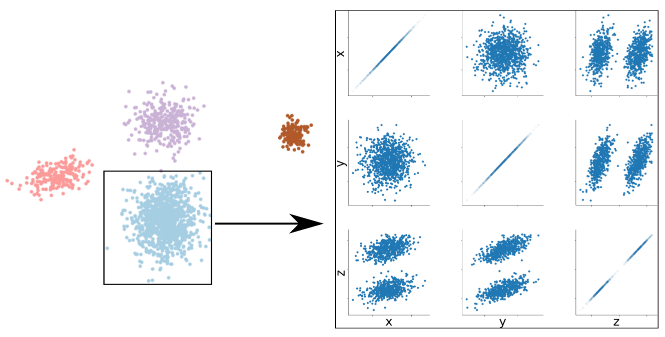
2.2 Density-Based Scatter and Parallel Coordinate Plots

Scatter and parallel coordinate plots can be used to visualize multivariate data. For large data, these item-based visualizations are challenged by overdraw and visual clutter. Instead of drawing discrete glyphs, density estimation methods reconstruct and visualize a continuous density of data values. For scatterplots, a simple form of density estimation is to draw individual points semi-transparently using alpha blending. Histograms and hexagonal binning are often employed to convey frequency information, but can lead to aliasing due to their discrete nature. The concept of histograms has also been extended to parallel coordinate space [1,3,34]. Although kernel density estimation would allow for an improved reconstruction of continuous density, it is computationally expensive. Splatterplots [29] perform kernel density estimation to avoid overdraw, but explicitly add representative outliers. We estimate density using GMMs and similarly support the explicit visualization of outliers.

In the field of scientific visualization, continuous scatter and parallel coordinate plots have been introduced [2,14] to construct density plots by considering the topology and interpolation of data samples in their spatial domain. Despite optimizations [13], this remains a computationally challenging approach that is unsuited for the interactive analysis of large-scale datasets.

2.3 Rendering 3D Gaussians

The encoding of scattered, unstructured, or large volumetric data using radial basis functions (RBF) has been an active research topic [5,17,18,47]. This involves the rendering of isotropic and anisotropic Gaussian kernels [16,20,33]. In detail, Zwicker et al. [49] discuss splatting of elliptical Gaussians by approximating the footprint after perspective projection. They extend their splatting approach by combining the reconstruction with a low-pass kernel, which could be similarly applied to our approach. In contrast to previous work, we derive a closed-form solution to integrate a Gaussian kernel along a ray. This enables us to efficiently splat large amounts of Gaussians without requiring expensive precomputation. Note that we consider three-dimensional Gaussians defined by a mean and covariance matrix, which makes the use of a view-independent look-up table infeasible. Additionally, we employ moment-based order-independent transparency [31] to address the short-comings of back-to-front splatting. Our method could also be employed during volume ray casting [23].



(a) Clustered data (b) Scatter plot matrix

Fig. 2: From a given clustering of the data (a), we model each cluster using combinations of low-dimensional distributions, similar to a scatter plot matrix (b).

3 Probabilistic Summaries

In this work, we describe the creation of probabilistic data summaries for multivariate, scattered data. We assume that the data is clustered into spatially coherent regions [8]. In Sect. 6, we discuss both domain specific and standard clustering techniques for scattered data. Similar to previous work, we employ Gaussian mixture models to represent data distributions in each cluster. However, these have not been applied to multivariate data. High-dimensional Gaussian mixture models require immense computational effort and due to the curse of dimensionality, there are not enough samples to cover a multi-dimensional space extensively.

Our approach is based on the observation that we do not require more than three data-dimensions at once to employ common interactive visual analysis techniques. In fact, the visualization of the spatial distribution is the only aspect considering correlations of three dimensions. Therefore, our approach is to only generate GMMs for the relevant combinations of dimensions. By default, these are all individual dimensions, all pairs of dimensions (cf. Fig. 2) and all vectorial attributes. As for high-dimensional GMMs, the storage cost grows quadratically with the number of dimensions. To better reason about our approach, we first introduce it more formally.

3.1 Data Model

Our data consists of $n \in \mathbb{N}$ samples. Each sample is associated with a position in 3D space, $m_v - 1 \in \mathbb{N}_0$ additional vectorial attributes and $m_s \in \mathbb{N}_0$ scalar attributes. Of course, the approach is also applicable to scattered data in 1D or 2D space. We denote the data for sample $i \in \{0, \ldots, n-1\}$ by:

- $v_{i,0} \in \mathbb{R}^{1\times 3}$ for the position,
- $v_{i,j} \in \mathbb{R}^{1\times 3}$ for vectorial attribute $j \in \{1, \ldots, m_v - 1\}$,
- $s_{i,j} \in \mathbb{R}$ for scalar attribute $j \in \{0, \ldots, m_s - 1\}$.

To define our probabilistic summaries, we concatenate all attributes for sample $i \in \{0, \ldots, n-1\}$ into a single vector with $m_u := 3m_v + m_s$ entries to enable linear indexing:

$$u_i := (v_{i,0}, v_{i,1}, \ldots, v_{i,m_v-1}, s_{i,0}, \ldots, s_{i,m_s-1}) \in \mathbb{R}^{1\times m_u}.$$

GMMs are generated for each given cluster $\mathbb{I} \subseteq \{0, \ldots, n-1\}$ and for each relevant combination of dimensions. First, we generate 1D models for each attribute $j \in \{0, \ldots, m_u - 1\}$:

$$(u_{i,j})_{i\in\mathbb{I}} \in \mathbb{R}^{|\mathbb{I}|\times 1}.$$

Then, we generate 2D models for each pair of dimensions $j,k \in \{0, \ldots, m_u - 1\}$ with $j < k$:

$$(u_{i,j}, u_{i,k})_{i\in\mathbb{I}} \in \mathbb{R}^{|\mathbb{I}|\times 2}.$$

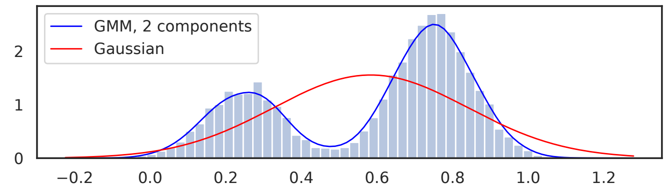


Fig. 3: To the distribution shown by the histogram, we have fitted a Gaussian (red) and a mixture of two Gaussians (blue). In this example, the GMM better models the data.

Finally, we generate 3D models for each vectorial attribute $j \in \{0, \ldots, m_v - 1\}$:

$$(v_{i,j})_{i\in\mathbb{I}} \in \mathbb{R}^{|\mathbb{I}|\times 3}.$$

Of course, the generation of GMMs for particular combinations of attributes may be skipped if the analysis of their mutual dependence is of no interest in a specific application.

Our probabilistic summary is the combination of all these low-dimensional GMMs for all clusters. They capture all information needed for common interactive visual analysis techniques but limit the analysis of higher dimensional correlations. By modeling only low-dimensional distributions, the curse of dimensionality does not apply. Ultimately, this limitation enables us to create reliable models of multivariate data.

3.2 Gaussian Mixture Models

We use GMMs because they offer a compact and efficient representation of the target distributions and have been employed successfully for modeling low-dimensional distributions in previous work [7, 8, 45]. In the following, we provide more details on our fitting procedure.

We generate a GMM for each combination of a cluster $\mathbb{I}$ and a relevant subset of attributes $\mathbb{J} \subseteq \{0, \ldots, m_u - 1\}$. As explained above, the number of attributes $|\mathbb{J}|$ is one, two, or three. A GMM is indexed by a pair $g := (\mathbb{I}, \mathbb{J})$ and consists of $n_g \in \mathbb{N}$ weighted Gaussians. Gaussian $l \in \{0, \ldots, n_g - 1\}$ is given by its weight $w_{g,l} > 0$, its mean $\mu_{g,l} \in \mathbb{R}^{|\mathbb{J}|}$ and its covariance $\Sigma_{g,l} \in \mathbb{R}^{|\mathbb{J}|\times|\mathbb{J}|}$. The density of a GMM at $p \in \mathbb{R}^{|\mathbb{J}|}$ is the weighted sum of the individual Gaussian densities (Fig. 3):

$$\rho_g(p) := \sum_{l=0}^{n_g-1} \frac{w_{g,l}}{\sqrt{|2\pi\Sigma_{g,l}|}} \exp\left(-\frac{(p-\mu_{g,l})^\mathsf{T}\Sigma_{g,l}^{-1}(p-\mu_{g,l})}{2}\right).$$

We compute the parameters of a GMM from a sequence of input samples with the expectation maximization (EM) procedure. This iterative method seeks maximum likelihood estimates of the model parameters. It alternates between an expectation step, which evaluates the log-likelihood of the input samples using the current parameters, and a maximization step, which computes the parameters by maximizing the expected log-likelihood found in the expectation step.

3.3 Fast Selection of GMM Components

The EM algorithm takes the number of Gaussian components n_g as input. With more components, the target distribution can be modeled better. However, too many components may not significantly improve the model, but increase the storage overhead. A fixed, arbitrary number of number of components is often used [7, 8, 12]. Similar to Wang et al. [45], we adaptively select the appropriate number of components, but propose approximations to significantly reduce the computational complexity.

We iteratively fit GMMs with an increasing number of components up to a user specified maximum and select the GMM with the best Bayesian information criterion (BIC) [38]. The BIC rewards a high likelihood over the training data and penalizes by the number of components. It is defined using the number of free parameters k_{GMM} in the GMM as

$$-2L_p + k_{\text{GMM}} \log|\mathbb{I}|,$$

where L_p denotes the maximized log-likelihood and k_{GMM} is given by

$$k_{\mathrm{GMM}} := n_g \left(\frac{|\mathbb{J}|(|\mathbb{J}|+1)}{2} + |\mathbb{J}| \right) + |\mathbb{J}| - 1.$$

The iterative computation of GMMs with different numbers of components is computationally challenging, especially for large clusters. To speed up the selection of the best n_g, we propose two approximations: First, we take a random subset $\mathbb{I}_{\mathbb{S}} \subset \mathbb{I}$ of our cluster whilst iteratively estimating the GMMs. After we have selected the best n_g based on the BIC, we recompute the GMM with n_g components for the whole cluster $\mathbb{I}$.

Second, after we have selected the number of components $\{n_0, \dots, n_{m_u-1}\}$ for all one-dimensional GMMs, we use them as lower and upper bounds for the two- and three-dimensional GMMs. In detail, for a subset of attributes $\mathbb{J} \subseteq \{0, \dots, m_u - 1\}$, we define the lower bound as

$$n^{\min}_{\mathbb{I},\mathbb{J}} := \min_{j \in \mathbb{J}} n_{\mathbb{I},\{j\}}$$

and an approximate upper bound as

$$n^{\max}_{\mathbb{I},\mathbb{J}} := \Pi_{j \in \mathbb{J}} n_{\mathbb{I},\{j\}}.$$

This implies that the higher-dimensional GMMs include at least the complexity of lower dimensions, whilst being bound by all combinations of all lower dimensional Gaussian components. In the supplementary material, we show that the bounds introduce no error, whilst the subsampling introduces a small error on our datasets.

Lastly, it is possible that some clusters contain only a small number of data samples. Although such a clustering may not seem optimal, it is quite likely to occur for scattered data. For very small clusters, e.g. $|\mathbb{I}| \leq 20$, fitting a GMM is problematic since the target distribution may be underdetermined. In this case, we fit a single Gaussian to these clusters.

4 Spatial Visualization

In this section, we discuss the visualization of the spatial density distribution. Although we could reconstruct the original data by drawing samples from the GMM of each cluster, this would require rendering a large amount of scattered data. Instead, we derive an efficient formulation to directly splat three-dimensional Gaussians. Additionally, we consider the application of a transfer function to a one-dimensional value distribution in each cluster.

4.1 Integrating Visibility for Gaussians

To render a trivariate Gaussian distribution, we integrate along a view ray $o + xd$ starting at $o \in \mathbb{R}^3$ in normalized direction $d \in \mathbb{R}^3$ with $x \in \mathbb{R}$. The Gaussian is given by its weight $w := w_{g,l}$, mean $\mu := \mu_{g,l} \in \mathbb{R}^3$ and covariance $\Sigma := \Sigma_{g,l} \in \mathbb{R}^{3\times 3}$. To derive a general solution, we integrate over $[a,b]$ by substituting the ray equation into the trivariate Gaussian distribution:

$$I(a,b) := \int_a^b \frac{1}{\sqrt{|2\pi\Sigma|}} \exp\left(-\frac{(o + xd - \mu)^{\mathsf{T}} \Sigma^{-1} (o + xd - \mu)}{2} \right) \mathrm{d}x.$$

Through integration by substitution (see the supplementary material), we obtain the following closed-form solution:

$$I(a,b) = c \frac{\sqrt{\pi}}{\sqrt{c_{d,d}}} \left[\frac{1}{2} \operatorname{erf}(y) \right]_{\sqrt{c_{d,d}}\left(a + \frac{c_{o,d}}{c_{d,d}}\right)}^{\sqrt{c_{d,d}}\left(b + \frac{c_{o,d}}{c_{d,d}}\right)}, \tag{1}$$

with

$$\begin{aligned} c_{d,d} &:= \frac{1}{2} d^{\mathsf{T}} \Sigma^{-1} d, \\ c_{o,d} &:= \frac{1}{2} (o - \mu)^{\mathsf{T}} \Sigma^{-1} d, \\ c &:= \frac{1}{\sqrt{|2\pi\Sigma|}} \exp\left(-\frac{1}{2} (o - \mu)^{\mathsf{T}} \Sigma^{-1} (o - \mu) + \frac{c_{o,d}^2}{c_{d,d}} \right). \end{aligned}$$

When integrating over all of $\mathbb{R}$, this result simplifies to

$$I(-\infty, \infty) = c \frac{\sqrt{\pi}}{\sqrt{c_{d,d}}}. \tag{2}$$

We could use this result inside a ray tracer, possibly with ray tracing GPUs [22]. It only has to identify relevant Gaussians per pixel, ray marching for integration becomes unnecessary. In the following, we discuss our approach using GPU rasterization, which works efficiently on commodity graphics hardware.

4.2 Back-to-Front Splatting

To splat scattered 3D Gaussians, we sort them from back-to-front based on their mean distance to the camera. Then, we integrate the Gaussians along the viewing direction using Equation 2. Integrating from $-\infty$ to ∞ is generally a reasonable approximation, but it is possible that we incorrectly evaluate a Gaussian if the camera is positioned within its support. Alternatively, we could employ Equation 1, but this is far more expensive and only gives a benefit in rare cases.

To render a single 3D Gaussian, we first compute the principal components of the distribution to fit a bounding box along the principal axes. By default, we limit the size of the bounding box in each dimension by 3 standard deviations. This box is then rasterized and for each resulting fragment, we integrate the Gaussian along the viewing direction in a fragment shader using Equation 2. Finally, we tone-map the resulting density (see Equation 4) to better convey the high-dynamic range.

We did experience some numerical issues with some of our datasets due to very large or small spatial and value domains. We were able to address these issues by switching to a more numerically stable eigen decomposition [10, Algorithm 8.2.3]. Lastly, we make use of the Cholesky decomposition to invert covariance matrices, which behaves robustly even for nearly singular matrices [43, p. 176].

4.3 Order-Independent Transparency

The splatting approach assumes that distributions do not overlap since this could lead to visible flickering between frames when the order changes. Depending on the clustering of the data, this assumption is not always acceptable. For this reason, we propose the use of an order-independent transparency (OIT) approach to avoid sorting semi-transparent Gaussians. Although, a large number of Gaussians are problematic for most OIT approaches, moment-based order-independent transparency (MBOIT) [31] is well-suited for this application. We introduce the steps of this method briefly.

MBOIT first accumulates moments of the optical depth in an additive rendering pass. These moments offer a compact representation and an efficient reconstruction of the transmittance function per view ray. Subsequently, a second additive rendering pass of all Gaussians composites the fragment colors using transmittance values reconstructed from the moments. We use three trigonometric moments in half-precision, which results in a total of 112 bits per pixel for the moments.

4.4 Uncertainty Transfer Function

Lastly, we discuss how to apply a transfer function to the distribution of an attribute $j \in \{n_0, \dots, n_{m_u-1}\}$ for each cluster $\mathbb{I}$. The one-dimensional value dimension is modeled separately from the cluster as a GMM $g = (\mathbb{I}, \{j\})$ with n_g components. For each cluster, we compute an expected color and opacity [37] by convolving the transfer function f with the value distribution:

$$E[f|g] := \int_{-\infty}^{\infty} f(p) \rho_g(p) \mathrm{d}p.$$

We insert the Gaussian mixture model into this equation and rearrange:

$$E[f|g] = \sum_{l=0}^{n_g - 1} w_{g,l} \int_{-\infty}^{\infty} f(p) \frac{1}{\sqrt{2\pi\sigma_{g,l}^2}} \exp\left(-\frac{(p - \mu_{g,l})^2}{2\sigma_{g,l}^2} \right) \mathrm{d}p. \tag{3}$$

We efficiently evaluate this equation by precomputing the integrand, which is simply a convolution of the transfer function with differently parametrized Gaussians. The resulting 2D lookup table thus depends on the transfer function and is parameterized by mean and variance.

5 Visual Analysis

Now that we are able to render the spatial distribution of our data, we move on to the visual exploration and analysis of additional data dimensions using our representation. This includes multiple views with brushing and linking coupled with a focus and context visualization to emphasize brushed values.

5.1 Density-Based Visualization

Prior work relies on sampling to create visualizations from modeled distributions. Although this is similarly possible with our data representation, see Fig. 11 (c) and (d), we focus on density-based visualizations. Since we already have an estimate of density in the form of our GMMs, we efficiently construct density-based visualizations that are costly to compute otherwise. To obtain the density, we evaluate and accumulate the Gaussian distributions from the GMMs in all clusters. Since the distribution in each cluster is normalized, we additionally weight each cluster $\mathbb{I}$ by the normalized number of samples it represents $\frac{1}{n}|\mathbb{I}|$.

5.1.1 Density Plots

To compute a density in 1D or 2D, we evaluate and accumulate the Gaussian distributions, see Fig. 1(b) and (c). Since this operation can be parallelized trivially, we make use of GPU acceleration. In the 1D case, we evaluate 1D Gaussians on the GPU and plot a probability density function. For a 2D plot, we render a quadrilateral for each Gaussian, evaluate the Gaussian for each fragment, and additively accumulate the results.

5.1.2 Parallel Coordinate Plots

Miller and Wegman [30] formulate parallel coordinate plots for bivariate Gaussian distributions. With this formulation, we splat the 2D distributions for each pair of consecutive dimensions in parallel coordinate space. Specifically, for each pair of axes we draw a quad for each Gaussian by truncating its support to three standard deviations. In the fragment shader, we evaluate the density in parallel coordinate space and additively blend the result with all other Gaussians. A density-based parallel coordinate plot is shown in Fig. 1(d).

5.1.3 Mapping Density

The density-based visualizations described above and the spatial visualization in Sect. 4 all produce a single density ρ per Gaussian and per pixel. For large datasets, this density will have a high dynamic range and needs to be mapped to an opacity between zero and one through a non-linear mapping [19]. We choose a mapping that interprets the density, scaled by a user-controllable parameter $\lambda > 0$, as optical depth. The resulting opacity is

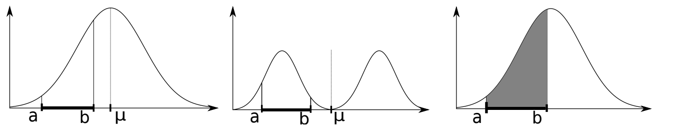
$$1 - \exp(-\lambda\rho). \tag{4}$$

With this mapping, multiplying the density of a Gaussian by an integer factor k produces the same result as rendering it k times with alpha blending, which is an intuitive behavior. At the same time, it retains detail even for large densities.

5.2 Brushing Distributions

To brush a value range of a dimension with our data representation and reflect this in all linked views, we use the clustering information. Although we could brush based on the cluster mean value, this is confusing and not very intuitive, especially when considering a dimension represented by multiple Gaussian components, see Fig. 4(a) and (b).

Feng et al. [9] discuss user interaction based on Gaussian distributions in the context of uncertainty visualization. We generalize their work and the concept of smooth brushing [6] to Gaussian mixture models. In detail, we compute the amount a cluster is in focus, the degree of interest, as the ratio between the integrand of the brushed regions of the GMM and the total area, see Fig. 4(c). For clusters that contain multiple Gaussian components, we compute the degree of interest as the weighted sum of all components.



(a) Brushing (mean) (b) Brushing (mean) (c) Brushing (distribution)

Fig. 4: Brushing of a value range $[a,b]$ applied to several distributions. Brushing only based on the mean value μ would lead to confusing results (a), especially if the distribution is represented by multiple Gaussian components (b). We compute the degree of interest of the brushing operation as the ratio between the integrand (gray) and the total area under the curve (c).

5.3 Time-Dependent Visualization

Brushing in different time steps is a powerful tool for the interactive exploration of time-dependent data [15], but is not practical for large datasets since all time steps have to be processed. Our compact data summaries enable us to interact with multiple time steps at once. We support this interaction in a time histogram [24] where we depict a time-series of a selected dimension as a series of 1D histograms, see Fig. 10.

If the clustering is fixed over time, we can trivially extend the brushing operation to time-dependent data. This is not possible when clusters change over time, e.g. merge together into larger, or split into smaller clusters. In this case, the relationship of clusters in different time steps has to be explicitly modeled and stored.

For brushing, we need to reassign degrees of interest from frame to frame. To this end, we transfer the degrees of interest to the individual samples uniformly and then reassign them to clusters. Say we have $n_t \in \mathbb{N}$ clusters $\mathbb{I}_{t,0},\dots,\mathbb{I}_{t,n_t-1} \subseteq \{0,\dots,n-1\}$ in frame t and analogously for frame $t+1$. The clusters in frame t have associated degrees of interest $d_{t,0},\dots,d_{t,n_t-1} \in [0,1]$. Then we define the degree of interest of cluster $k \in \{0,\dots,n_{t+1}-1\}$ in frame $t+1$ as

$$d_{t+1,k} := \sum_{l=0}^{n_t-1} \frac{|\mathbb{I}_{t,l} \cap \mathbb{I}_{t+1,k}|}{|\mathbb{I}_{t+1,k}|} d_{t,l} \in [0,1].$$

The quotient in this sum is the fraction of samples in cluster $\mathbb{I}_{t+1,k}$ that was part of cluster $\mathbb{I}_{t,l}$ in the previous frame. Interest is inherited from the cluster in the previous frame in proportion to that quotient. Note that this method defines a simple linear transform. There is no need to consider all samples at run time. Instead, the transfer coefficients for the degrees of interest can be precomputed and stored in a sparse matrix.

5.4 Uncertainty Visualization

We introduce an error estimate to convey the uncertainty of the data summaries. By computing and storing an error for each cluster, we are able to visualize the uncertainty interactively during the visual analysis and to support brushing and linking. Prior work measures the error directly between the density of the Gaussian mixture model and the original data. However, this is not robust and suffers from aliasing due to the necessary use of histograms. Instead, we define the error between a Gaussian mixture model and the samples of a cluster $\mathbb{I}$ for a dimension $j \in \{0,\dots,m_u-1\}$ similar to common statistical goodness of fit tests. In detail, we compute the empirical cumulative distribution functions (CDF) of the data samples

$$F_{\mathbb{I}}(p) := \frac{1}{|\mathbb{I}|} \sum_{i\in\mathbb{I}} \begin{cases} 1 & \text{if } u_{i,j} \le p, \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

and compare it to the CDF F_g of the Gaussian mixture model using the Wasserstein distance [35]:

$$W(F_{\mathbb{I}}, F_g) := \int_{-\infty}^{\infty} \left|F_{\mathbb{I}}(p) - F_g(p)\right| \mathrm{d}p. \tag{6}$$

To visualize the Wasserstein distance, we show it together with the CDF, cf. Fig. 8 (b). A high Wasserstein distance consequently indicates a high uncertainty of the data model.

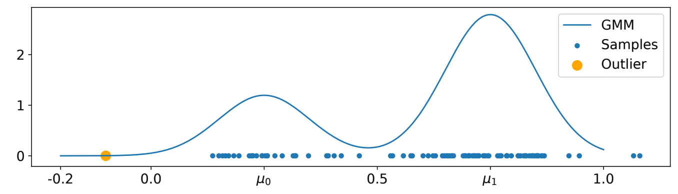
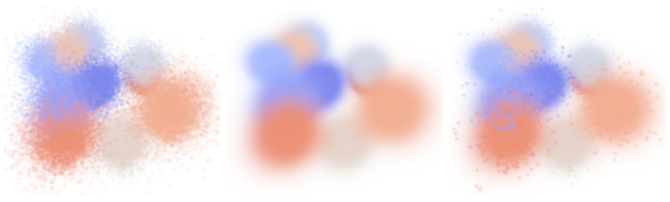


Fig. 5: To rank samples by their outlyingness, we evaluate the Mahalanobis distance to the closest Gaussian. This measures how many standard deviations a sample is away from the mean of the closest Gaussian.



(a) Synthetic data (b) GMMs (c) GMMs and outliers

Fig. 6: Rendering of Gaussians from the synthetic dataset using kernel density estimation (a), with our data model (b), and with 2% of outliers (c).

5.5 Level of Detail and Outliers

By design, our representation is a simplified model of the data. During the exploration and analysis process, a scientist might want to investigate a subset of the data more closely. For this purpose, we substitute brushed clusters by their original data values. To integrate the data distributions into our frequency-based views, we perform kernel density estimation using Gaussian kernels. We can thus avoid differentiating between the modeled and original data distributions.

Moreover, outliers, i.e. isolated samples in regions of low density, tend to get lost in density-based visualizations [29, 34]. To explicitly add outliers to our visualizations, we sort all samples in a cluster in a preprocess according to a measure of outlyingness. Although any measure between a sample and a GMM could be used, we employ the Mahalanobis distance [28] to the closest Gaussian component. This effectively measures how many standard deviations a sample is away from the mean of the closest Gaussian, see Fig. 5. To visualize outliers, we then take a fixed percentage p_o of outliers from a cluster $\mathbb{I}$ by loading the first $p_o|\mathbb{I}|$ samples. Fig. 6 (c) shows a spatial visualization with 2% of outliers.

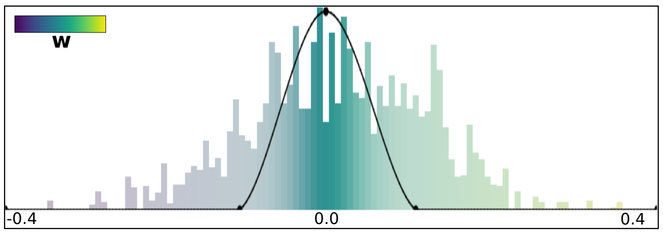
6 Evaluation

In this section, we apply our approach to a synthetic and three real-world datasets. Additional results can be found in the supplementary material.

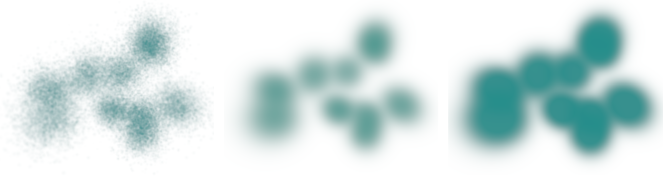
6.1 Synthetic Data

We first apply our approach to a small synthetic dataset consisting of 10 clusters from a total of 100000 points. The dataset contains 9 dimensions. The three spatial dimensions in each cluster are normally distributed, but 10 % of the points are distributed uniformly to add noise to the distributions. Fig. 6 (a) shows this dataset. The 3D Gaussians are shown in Fig. 6 (b) and in (c) where we explicitly add 2% of outliers from all clusters.

We compare the uncertainty transfer function to a 1D transfer function based on mean values in Fig. 7. In (a), we set the opacity of the transfer function, where the peak coincides with the mean value. The synthetic dataset in (b) shows the resulting rendering. For our data summaries in (c), the opacity is similarly reduced. This is due to the uncertainty transfer function since it computes the expected opacity with respect to the value distribution. In comparison, the opacity of the Gaussians in (d) using a mean transfer function does not change since the opacity of the mean value is still set to opaque in (a). Thus, changing any of the opacity (or color) values of the transfer function has no influence except if the mean value is changed. An alternative would be the use of a 2D transfer function [21] that offers increased control over the classification, but complicates user interaction.

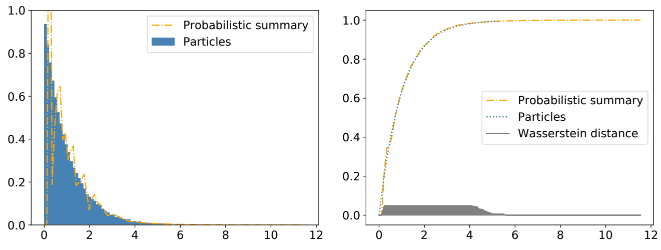


(a) Transfer function



(b) Synthetic data (c) Expected opacity (d) Mean opacity

Fig. 7: We set the opacity of the transfer function (a) to visualize the synthetic dataset (b). Our uncertainty transfer function (c) computes the expected opacity (i.e. the integral of the opacity curve), while a 1D transfer function based on the mean value (d) sets all Gaussians to opaque.



(a) Probability density function (b) Cumulative distribution function

Fig. 8: The exponentially distributed dimension (a) is hard to model using Gaussian components. The cumulative distribution function in (b) conveys the error to the user.

In Fig. 8 (a) we illustrate an exponentially distributed dimension of the dataset, which is difficult to model using only Gaussian components. The cumulative distribution function shown in (b), illustrates this error as measured by the Wasserstein distance. By quantifying the error, we can decide if this error is acceptable, or brush and use the level-of-detail mechanism to directly load a subset of the data with a high error.

6.2 Cosmological Data

The Illustris simulation [32] is a large-scale cosmological hydrodynamical simulation of galaxy formation that aims to predict both dark and baryonic components of matter. In detail, the dynamics of dark matter and gas are simulated with the quasi-Lagrangian code AREPO [41], which employs an unstructured Voronoi tessellation of the domain. After simulation, only the center points of the Voronoi cells are kept and are referred to as particles. Since the simulation has been run in different resolutions and we want to show both dark and baryonic matter, we discuss multiple separate datasets as shown in Table 1. We compare the data based on the 100th time step without descendant or ancestor information.

The Illustris datasets have been clustered into halos using a domain specific approach. The sizes of clusters are extremely irregular and range in between a single particle and up to millions of particles per cluster. Since we cannot fit a GMM to very small clusters, we fit a single Gaussian for clusters of size below 20.

Table 1: Overview of the cosmological data from the Illustris simulation.

Dataset	# Dimensions	# Particles	# Clusters	Data size	Summaries	GMM comp.	Wasserstein dist.
Illustris-3 Gas	15	16,039,182	110,000	2 GB	200 MB	2.15 ± 1.60	1.77×10^{-6}
Illustris-2 DM	7	319,324,195	841,639	11.3 GB	617 MB	1.54 ± 1.22	3.10×10^{-7}
Illustris-1 DM	6	2,635,739,426	5,352,571	72.2 GB	1.5 GB	1.13 ± 0.38	4.56×10^{-8}

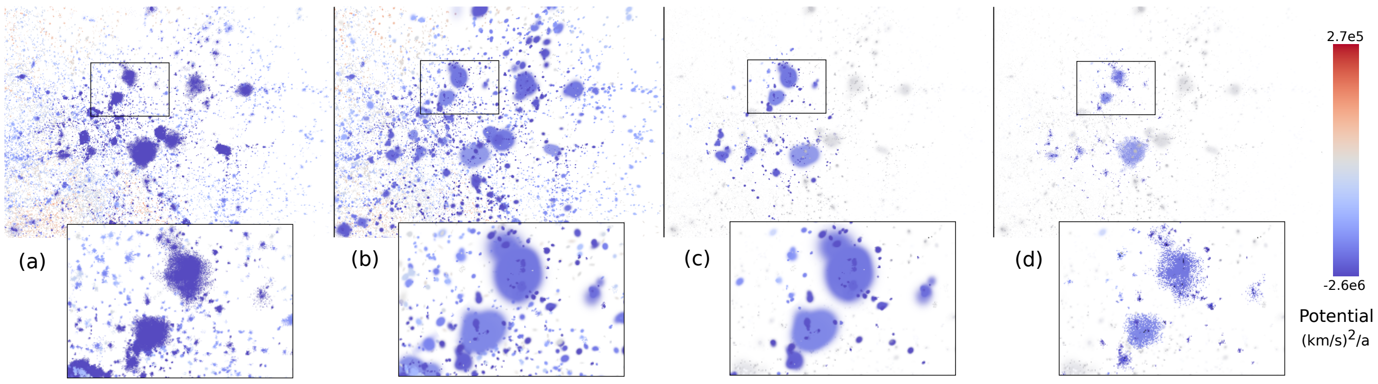
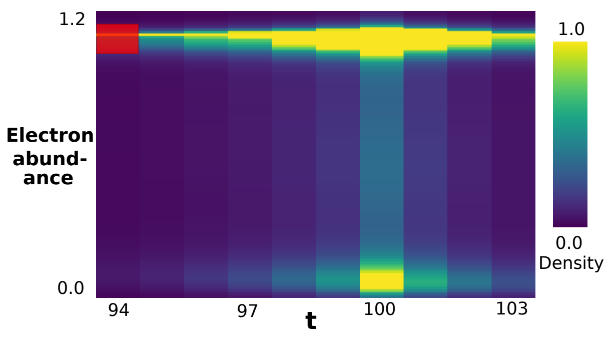
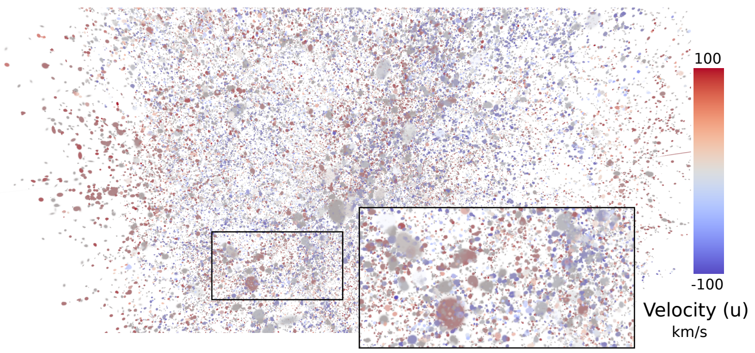


Fig. 9: The Illustris-3 Gas dataset rendered by splatting particles (a) and 3D Gaussians (b). In (c) we have brushed a region and clusters that are not in focus are shown in a desaturated gray. We load the original particle data of the brushed region and render them together with the context (d).

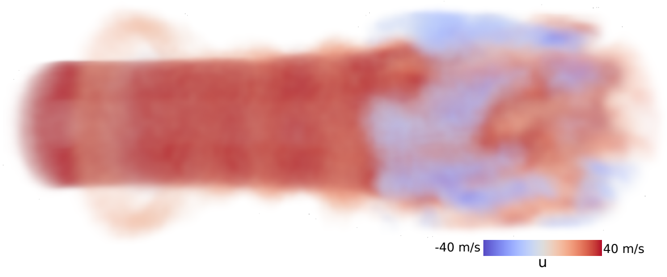


(a) Time histogram

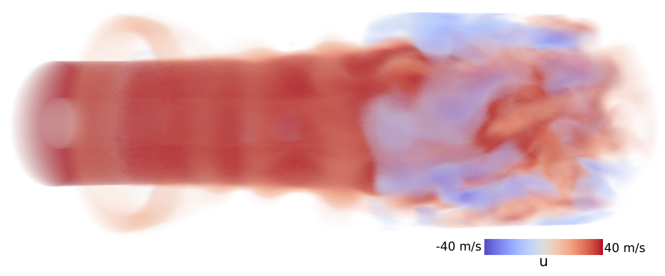


(b) Brushed clusters in the 100th time step

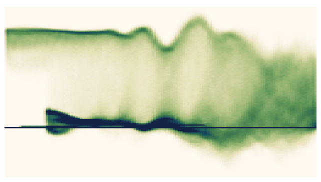
Fig. 10: A time-histogram of electron abundance in the Illustris-3 Gas dataset is shown in (a). We have brushed (red) in the 94th time step, which affects all linked views in the current, 100th time step. The spatial visualization that highlights the brushed values in the 100th time step is shown in (b). Note that Gaussians are shown as saturated or desaturated, depending on how much they are in focus.



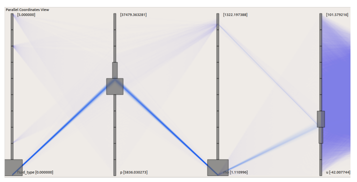
(a) Splatting 3D Gaussians (*k*-means 32.000)



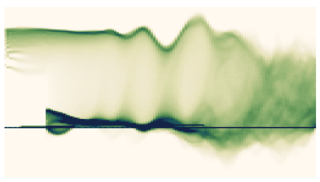
(b) Splatting all particles



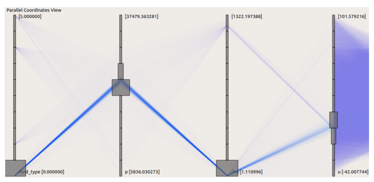
(c) Histogram from samples



(d) PCP from samples



(e) Histogram from all particles



(f) PCP from all particles

Fig. 11: Visualization of a spray nozzle using our approach with the *k*-means 32.000 clustering by splatting 3D Gaussians (a) and by drawing samples from the GMMs to create a 2D histogram (c) and a parallel coordinate plot (d). In (b), (e), and (f) the corresponding visualizations using the original SPH particle data are shown.

Table 2: Overview of the spray nozzle dataset. We show the absolute summary size, relative to the original data size, the average number of GMM components, and the average Wasserstein distance.

# Clusters	Summaries	Rel. size	GMM comp.	Wasserstein dist.
2,000	6.7 MB	0.006%	1.7 ± 1.19	5.54×10^{-5}
8,000	19.9 MB	0.017%	1.4 ± 0.85	1.57×10^{-5}
32,000	47.5 MB	0.036%	1.2 ± 0.61	4.64×10^{-6}

With our approach, we are able to interactively visualize and explore these massive datasets that might not even fit into memory otherwise. Fig. 1 shows several interactive, linked views of the Illustris-1 dataset. We have brushed the x-, y-, and z-axis in the parallel coordinate plot (d). The brushed regions (green) are then highlighted in red in all other views. The density-based views are free of clutter and clearly show trends and correlations between the dimensions. For example, the parallel coordinate plot in (d) indicates that the brushed values have velocity components that are distributed around zero and are linearly correlated. The spatial visualization depicts 5.3 million clusters that we render and navigate interactively.

Fig. 9 compares our probabilistic summaries with the original particle data of Illustris-3 Gas. Note that the interactive visualization of Illustris-1 and 2 is not possible on our system due to their data sizes. Although we clearly miss some details in the spatial visualization, we still manage to convey the general structure of the data and the distribution of color-mapped values. Whilst sorting and rendering all 16 million particles as isotropic Gaussians takes 61 ms on our system, the clusters require only 2 ms. Note that for this dataset, the $110,000$ clusters are represented by a total of $357,512$ Gaussians in 3D. In Fig. 9(c), we have brushed a spatial region on the right side which is consequently put into focus. In (d) we have loaded the original particle data of the brushed clusters. Note that all of the linked views are also updated by this operation. Since we only load an additional $240,000$ particles, the interactive visualization still takes only 3.8 ms to render. Although other forms of level of detail are possible, this is a powerful way to drill-down from an overview to a detailed view of the data.

Since the clusters split and merge over time in this dataset, we have precomputed the transfer coefficients for time-dependent visualizations. Fig. 10 depicts a time-histogram of a selected attribute over several time steps. We have brushed the 94th time step, which is reflected in all views in the current, 100th time step. The spatial visualization in (b) highlights those brushed values. Due to our brush, mostly smaller clusters are in focus. This brushing and linking over time thus allows exploring the selected dimension and its time-dependent behavior. With our data summaries, we can compute the whole time-histogram in just under a second. In comparison, computing a time-histogram from the particle data takes 34 s and is severely I/O limited since the complexity scales with respect to the number of particles instead of the amount of clusters.

6.3 Spray Nozzle

We have applied our technique to a smoothed particle hydrodynamics (SPH) dataset of a fuel spray nozzle simulation [4]. In the context of renewable energy production, biomass is converted into fuel by a gasification process. The quality of the spray is analyzed since it is critical for the efficiency of the gasification. However, the size of the time-dependent data prevents the usage of common interactive visual analysis techniques. In detail, the dataset contains about 43 million particles per time step. Each particle contains a position, velocity, pressure, density, and fluid type for a total of 9 separate dimensions. The fluid type describes four different categories, including fluid, gas, and two types of boundaries.

We have partitioned the data using a k-means clustering based on the spatial position, fluid type, and velocity magnitude. Table 2 shows the data size reduction and average number of GMM components for different numbers of clusters. For this dataset, we fix the maximum number of GMM components to 6. The size of the data summaries increases with the number of clusters. At the same time, the average number of GMM components decreases. This shows that the number of GMM components adapts to the less complex clusters. Moreover, the average Wasserstein distance is reduced for a larger number of clusters.

Table 3: Overview of the different clustering procedures of the Hurricane Isabel dataset. We show the resulting absolute and relative data size and the average Wasserstein distance.

Model	Clustering	# Clusters	Summaries	Rel. size	Wasserstein dist.
Our	Blocks	1,000	12.1 MB	1.4%	1.20×10^{-4}
HD	Blocks	1,000	5.2 MB	0.6%	1.21×10^{-4}
Our	Blocks	8,000	82.6 MB	9.0%	1.58×10^{-5}
HD	Blocks	8,000	34.7 MB	4.8%	1.56×10^{-5}
Our	Blocks	16,000	146.8 MB	14.8%	8.49×10^{-6}
HD	Blocks	16,000	50.0 MB	5.1%	8.24×10^{-6}
Our	k-means	1,000	12.1 MB	1.4%	1.23×10^{-4}
HD	k-means	1,000	5.4 MB	0.6%	1.26×10^{-4}
Our	k-means	8,000	80.5 MB	8.8%	1.66×10^{-5}
HD	k-means	8,000	33.8 MB	3.7%	1.57×10^{-5}
Our	k-means	16,000	143.9 MB	14.7%	8.75×10^{-6}
HD	k-means	16,000	48.5 MB	5.0%	8.01×10^{-6}

Fig. 11 depicts several visualizations created from our representation and from the original SPH data. The spatial visualizations in (a) and (b) depict velocity in u-direction. Our approach does lose some details, especially on the finer structures on the right side of the cylindrical domain. Since item-based visualizations of 43 million particles suffer from strong overdraw and visual clutter, density-based visualizations are preferable for this dataset. These are fast and efficient to create using our representation that is already an estimate of density. The 2D histogram in (b) and the parallel coordinate plot in (c) have been created from samples drawn from the GMMs. Compared to the reference plots in (e) and (f), we achieve nearly identical results. Moreover, it is possible to vary the number of samples, which could be used to create less cluttered visualizations, e.g. for scatter and parallel coordinate plots.

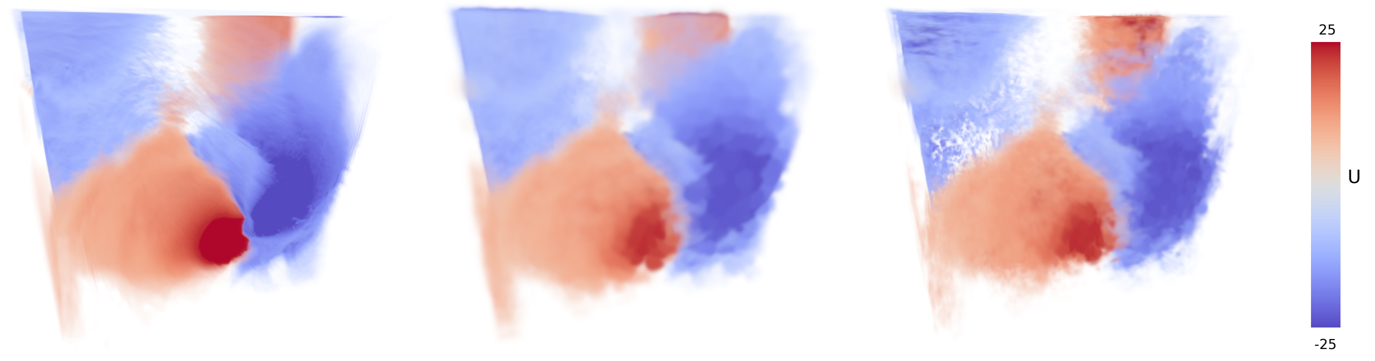
We represent the fluid type, i.e. the categorical dimension, by interpreting it as a scalar dimension. This is possible since the data only consists of four fluid types that we model using an appropriate number of Gaussian components. We could have increased the maximum number of components for all marginal distributions containing a categorical dimension, but this was not necessary for this dataset. Although a small number of categories is common in multiphase fluid simulations, in general, representing categorical dimensions with GMMs does not scale.

6.4 Hurricane Isabel

The Hurricane Isabel dataset is an atmospheric simulation from the IEEE Visualization Contest 2004, produced by the Weather Research and Forecast (WRF) model. Besides an implicit spatial position and a velocity vector, the time-dependent dataset contains 9 additional scalar quantities on a uniform grid of size $500 \times 500 \times 100$. Since we formulated our approach for scattered data and did not consider the important but special case of gridded data, we disregard the topology of the dataset.

To apply our approach, we either define clusters through uniform blocks or apply k-means clustering based on the spatial position and velocity magnitude. Both clustering procedures require a fixed number of clusters as input. Independent from the clustering, we always store 3D distributions for the spatial position and the velocity vector and compute the respective 2D marginal distribution from these. Apart from that, we model and store all pairwise 2D distributions and all 15 one-dimensional distributions. Additionally, we compare our representation to modeling each cluster with a high-dimensional Gaussian mixture model.

Table 3 shows the data summaries we have created with both clustering procedures, with different cluster sizes, with our approach and using high-dimensional Gaussian mixture models. We have chosen a maximum number of 6 GMM components for the low-dimensional



(a) Original data (b) Our low-dim. GMMs (c) High-dim. GMMs

Fig. 12: Visualization of wind speed from west to east (U) in the Hurricane Isabel data by splatting the original data (a), with the k-means 16.000 clustering of the low-dimensional model (b), and the high-dimensional model (c).

Table 4: Performance of visualizations with our data summaries.

Dataset	Splatting		PCP	Density	Brushing
	Sorting	OIT		(x,u)	
Illustris-3 Gas	3.9ms	4.3ms	439ms	1.0ms	4ms
Illustris-2 DM	31ms	14ms	421ms	3.6ms	28ms
Illustris-1 DM	196ms	28ms	1241ms	11.2ms	160ms
Hurricane Isabel k=8000	4.6ms	20ms	99ms	1.1ms	2ms
Spray Nozzle k=8000	4.4ms	23ms	47ms	1.4ms	2ms

Table 5: Measurements of the data summary preprocessing.

Dataset	Our GMMs	Low-dim. GMMs	High-dim. GMMs
Hurricane Isabel k=1000	2h 54m	9h 31m	42h 55m
Spray Nozzle k=2000	1h 43m	8h 13m	14h 51m

and 32 for the high-dimensional models to achieve a comparable quality. Note that creating the high-dimensional model took nearly 43 hours, cf. Table 5. Both approaches can model the data well even though some dimensions are quite challenging. The high-dimensional model performs surprisingly well for this dataset, considering the dimensionality, which is due to high correlations in the dimensions. The low-dimensional representation requires more storage since it cannot make use of these higher-dimensional correlations. In both cases, the two clustering procedures lead to similar results.

Fig. 12 shows a visualization of wind speed from west to east, i.e. u-velocity, by splatting the original data and with our approach. Although our representation loses some details, it conveys the major features of the dataset. Whilst the low-dimensional representation models the spatial position separately, the high-dimensional GMM takes correlations between all dimensions into account. The marginal distribution of the spatial positions is thus also influenced by the other dimensions, which leads to the artifacts in Fig. 12(c). This reduces trust in the high-dimensional model since it is unclear if these correlations actually exist in the data or not. Lastly, the high-dimensional model contains over five times the amount of Gaussian components, which increases the complexity of all visualizations. In comparison, our representation consists of low-dimensional models that are easier to understand and more robust.

6.5 Performance

Our evaluations were performed on an Intel i7-6700 with 32 GB of system memory and an NVIDIA GTX 1080 Ti graphics card providing 11 GB of video memory. For GPU acceleration, we make use of both CUDA for general purpose computations and OpenGL for rendering. For our spatial visualization, we have used a screen resolution of 1920×1080. The resolution of our 2D density plots was 200×200 and 800×300 for the parallel coordinate plot (PCP).

Timings for several visualizations are shown in Table 4. In general, our prototype allows interactive navigation and creation of all visualizations introduced above. The Illustris 1 and 2 datasets are the most demanding, due to the large number of clusters. Note that the performance of our approach scales with the number of clusters and Gaussian components, not the original data size. The order-independent transparency (OIT) approach performs very well on the cosmological datasets compared to the back-to-front splatting using sorting. Note that the speed varies depending on the number of covered pixels. The sorting approach is faster on the smaller and spatially more compact datasets.

We create our probabilistic data summaries in a preprocessing step using the Python scikit-learn library. This process is trivial to parallelize since all time steps, clusters, and distributions can be processed independently. Due to inherent restrictions imposed by our Python prototype, an implementation in a native language is expected to be significantly faster. The measurements for our prototype are shown in Table 5. Our fast GMM component estimation (Sect. 3.2) leads to a significant speedup. In the supplementary material, we show that a slight error is introduced by this approximation. Lastly, computing high-dimensional GMMs requires significantly more preprocessing time, making it unsuited for use in practice. Note that our approximations for a fast estimation of GMM components cannot be used for the high-dimensional data.

7 Conclusion

In this paper, we introduce probabilistic data summaries for multivariate scattered data. They enable the interactive visual analysis of large datasets that would not be possible otherwise due to limitations of memory or compute. Although our data representation is a simplified model of the data, we inform the user about this uncertainty and present a level-of-detail and outlier visualization for more detailed investigations.

The core insight of our approach is that we only have to model combinations of low-dimensional distributions for visual analysis, which avoids the complexity of modeling high-dimensional distributions. Although the data must be clustered, we do not make any restrictive assumptions about the clustering procedure. In fact, our evaluation shows that the impact of the clustering on the quality of the representation is less pronounced than expected and is largely offset by the adaptive modeling of GMMs.

In the future, we want to improve the scalability of our approach even further by adding a level-of-detail approach based on a hierarchical clustering of the data. By interactively selecting the appropriate detail, it should be possible to interactively explore massive datasets even on mobile devices and seamlessly scale up to powerful workstations.

Acknowledgments

The Spray Nozzle dataset is due to the Institute of Thermal Turbomachinery (ITS) at the Karlsruhe Institute of Technology (KIT). The Hurricane Isabel data is courtesy of NCAR, and the U.S. National Science Foundation (NSF).

References

[1] A. O. Artero, M. C. F. de Oliveira, and H. Levkowitz. Uncovering clusters in crowded parallel coordinates visualizations. In *IEEE Symposium on Information Visualization*, pp. 81–88, 2004. doi: 10.1109/INFVIS.2004.68

[2] S. Bachthaler and D. Weiskopf. Continuous scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1428–1435, 2008. doi: 10.1109/TVCG.2008.119

[3] J. Blaas, C. Botha, and F. Post. Extensions of parallel coordinates for interactive exploration of large multi-timepoint data sets. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1436–1451, 2008. doi: 10.1109/TVCG.2008.131

[4] G. Chaussonnet, S. Braun, T. Dauch, M. Keller, J. Kaden, C. Schwitzke, T. Jakobs, R. Koch, and H.-J. Bauer. SPH simulation of a twin-fluid atomizer operating with a high viscosity liquid. *14th Triennial International Conference on Liquid Atomization and Spray Systems*, 2018.

[5] C. S. Co, B. Heckel, H. Hagen, B. Hamann, and K. I. Joy. Hierarchical clustering for unstructured volumetric scalar fields. In *Proceedings of the 14th IEEE Visualization*, p. 43, 2003. doi: 10.1109/VISUAL.2003.1250389

[6] H. Doleisch and H. Hauser. Smooth brushing for focus+context visualization of simulation data in 3D. In *Journal of WSCG*, pp. 147–154, 2001.

[7] S. Dutta, C. M. Chen, G. Heinlein, H. W. Shen, and J. P. Chen. In situ distribution guided analysis and visualization of transonic jet engine simulations. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):811–820, 2017. doi: 10.1109/TVCG.2016.2598604

[8] S. Dutta, J. Woodring, H. W. Shen, J. P. Chen, and J. Ahrens. Homogeneity guided probabilistic data summaries for analysis and visualization of large-scale data sets. In *IEEE Pacific Visualization Symposium*, pp. 111–120, 2017. doi: 10.1109/PACIFICVIS.2017.8031585

[9] D. Feng, L. Kwock, Y. Lee, and R. Taylor. Matching visual saliency to confidence in plots of uncertain data. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):980–989, 2010.

[10] G. H. Golub and C. F. V. Loan. *Matrix Computations*. The Johns Hopkins University Press, 1993.

[11] S. Hazarika, A. Biswas, and H. W. Shen. Uncertainty visualization using copula-based analysis in mixed distribution models. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):934–943, 2018. doi: 10.1109/TVCG.2017.2744099

[12] S. Hazarika, S. Dutta, H. Shen, and J. Chen. CoDDA: A flexible copula-based distribution driven analysis framework for large-scale multivariate data. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):1214–1224, 2019. doi: 10.1109/TVCG.2018.2864801

[13] J. Heinrich, S. Bachthaler, and D. Weiskopf. Progressive splatting of continuous scatterplots and parallel coordinates. *Computer Graphics Forum*, 30(3):653–662, 2011. doi: 10.1111/j.1467-8659.2011.01914.x

[14] J. Heinrich and D. Weiskopf. Continuous parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1531–1538, 2009. doi: 10.1109/TVCG.2009.131

[15] H. Hochheiser and B. Shneiderman. Dynamic query tools for time series data sets: Timebox widgets for interactive exploration. *Information Visualization*, 3(1):1–18, 2004. doi: 10.1057/palgrave.ivs.9500061

[16] W. Hong, N. Neophytou, K. Mueller, and A. Kaufman. Constructing 3D elliptical Gaussians for irregular data. In *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*, pp. 213–225. Springer Berlin Heidelberg, 2009. doi: 10.1007/b106657_11

[17] Y. Jang, R. P. Botchen, A. Lauser, D. S. Ebert, K. P. Gaither, and T. Ertl. Enhancing the interactive visualization of procedurally encoded multifield data with ellipsoidal basis functions. *Computer Graphics Forum*, 25(3):587–596, 2006. doi: 10.1111/j.1467-8659.2006.00978.x

[18] Y. Jang, M. Weiler, M. Hopf, J. Huang, D. S. Ebert, K. P. Gaither, and T. Ertl. Interactively visualizing procedurally encoded scalar fields. In *Proceedings of the Sixth Joint Eurographics - IEEE TCVG Conference on Visualization*, pp. 35–44, 2004. doi: 10.2312/VisSym/VisSym04/035-044

[19] J. Johansson, P. Ljung, M. Jern, and M. Cooper. Revealing structure within clustered parallel coordinates displays. In *IEEE Symposium on Information Visualization*, pp. 125–132, 2005. doi: 10.1109/INFVIS.2005.1532138

[20] D. Juba and A. Varshney. Modelling and rendering large volume data with Gaussian radial basis functions. *University of Maryland, Technical Report No. UMIACS-TR-2007-22*, 2007.

[21] J. Kniss, G. Kindlmann, and C. Hansen. Multidimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):270–285, 2002. doi: 10.1109/TVCG.2002.1021579

[22] A. Knoll, R. K. Morley, I. Wald, N. Leaf, and P. Messmer. *Efficient Particle Volume Splatting in a Ray Tracer*, chap. 29, pp. 533–541. Apress, 2019. doi: 10.1007/978-1-4842-4427-2_29

[23] A. Knoll, I. Wald, P. Navratil, A. Bowen, K. Reda, M. E. Papka, and K. Gaither. RBF volume ray casting on multicore and manycore CPUs. *Computer Graphics Forum*, 33(3):71–80, 2014. doi: 10.1111/cgf.12363

[24] R. Kosara, F. Bendix, and H. Hauser. Time histograms for large, time-dependent data. In *Proceedings of the Sixth Joint Eurographics-IEEE TCVG conference on Visualization*, pp. 45–54, 2004.

[25] O. D. Lampe and H. Hauser. Interactive visualization of streaming data with kernel density estimation. In *IEEE Pacific Visualization Symposium*, pp. 171–178, 2011. doi: 10.1109/PACIFICVIS.2011.5742387

[26] G. Li, J. Xu, T. Zhang, G. Shan, H. Shen, K. Wang, S. Liao, and Z. Lu. Distribution-based particle data reduction for in-situ analysis and visualization of large-scale n-body cosmological simulations. In *IEEE Pacific Visualization Symposium*, pp. 171–180, 2020. doi: 10.1109/PacificVis48177.2020.1186

[27] S. Liu, J. A. Levine, P. T. Bremer, and V. Pascucci. Gaussian mixture model based volume visualization. In *IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, pp. 73–77, 2012. doi: 10.1109/LDAV.2012.6378978

[28] P. C. Mahalanobis. On the generalized distance in statistics. National Institute of Science of India, 1936.

[29] A. Mayorga and M. Gleicher. Splatterplots: Overcoming overdraw in scatter plots. *IEEE Transactions on Visualization and Computer Graphics*, 19(9):1526–1538, 2013. doi: 10.1109/TVCG.2013.65

[30] J. J. Miller and E. J. Wegman. Construction of line densities for parallel coordinate plots. *Computing and Graphics in Statistics*, 36:107–123, 1991.

[31] C. Münstermann, S. Krumpen, R. Klein, and C. Peters. Moment-based order-independent transparency. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 1(1):1–20, 2018. doi: 10.1145/3203206

[32] D. Nelson, A. Pillepich, S. Genel, M. Vogelsberger, V. Springel, P. Torrey, V. Rodriguez-Gomez, D. Sijacki, G. F. Snyder, B. Griffen, F. Marinacci, L. Blecha, L. Sales, D. Xu, and L. Hernquist. The Illustris simulation: Public data release. *Astronomy and Computing*, 13:12–37, 2015. doi: 10.1016/j.ascom.2015.09.003

[33] N. Neophytou, K. Mueller, K. T. McDonnell, W. Hong, X. Guan, H. Qin, and A. Kaufman. GPU-accelerated volume splatting with elliptical RBFs. In *Proceedings of the Eighth Joint Eurographics / IEEE VGTC Conference on Visualization*, pp. 13–20, 2006. doi: 10.2312/VisSym/EuroVis06/013-020

[34] M. Novotny and H. Hauser. Outlier-preserving focus+context visualization in parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):893–900, 2006. doi: 10.1109/TVCG.2006.170

[35] V. M. Panaretos and Y. Zemel. Statistical aspects of Wasserstein distances. *Annual Review of Statistics and Its Application*, 6(1):405–431, 2019. doi: 10.1146/annurev-statistics-030718-104938

[36] T. Rapp, C. Peters, and C. Dachsbacher. Void-and-cluster sampling of large scattered data and trajectories. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):780–789, 2020. doi: 10.1109/TVCG.2019.2934335

[37] E. Sakhaee and A. Entezari. A statistical direct volume rendering framework for visualization of uncertain data. *IEEE Transactions on Visualization and Computer Graphics*, 23(12):2509–2520, 2017. doi: 10.1109/TVCG.2016.2637333

[38] G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[39] R. Sicat, J. Krüger, 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. doi: 10.1109/TVCG.2014.2346324

[40] B. W. Silverman. *Density estimation for statistics and data analysis*. Routledge, 2018.

[41] V. Springel. E pur si muove: Galilean-invariant cosmological hydrodynamical simulations on a moving mesh. *Monthly Notices of the Royal Astronomical Society*, 401:791–851, 2010. doi: 10.1111/j.1365-2966.2009.15715.x

[42] 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 *IEEE Symposium on Large Data Analysis and Visualization*, pp. 23–30, 2011. doi: 10.1109/LDAV.2011.6092313

[43] L. Trefethen and D. Bau. *Numerical Linear Algebra*. Society for Industrial and Applied Mathematics, 1997.

[44] K. Wang, J. Xu, J. Woodring, and H. Shen. Statistical super resolution for data analysis and visualization of large scale cosmological simulations. In *IEEE Pacific Visualization Symposium*, pp. 303–312, 2019. doi: 10.1109/PacificVis.2019.00043

[45] K. C. Wang, K. Lu, T. H. Wei, N. Shareef, and H. W. Shen. Statistical visualization and analysis of large data using a value-based spatial distribution. In *IEEE Pacific Visualization Symposium*, pp. 161–170, 2017. doi: 10.1109/PACIFICVIS.2017.8031590

[46] G. H. Weber and H. Hauser. Interactive visual exploration and analysis. In *Scientific Visualization*, pp. 161–173. Springer London, 2014. doi: 10.1007/978-1-4471-6497-5_15

[47] M. Weiler, R. Botchen, S. Stegmaier, T. Ertl, J. Huang, Y. Jang, D. S. Ebert, and K. P. Gaither. Hardware-assisted feature analysis and visualization of procedurally encoded multifield volumetric data. *IEEE Computer Graphics and Applications*, 25(5):72–81, 2005. doi: 10.1109/MCG.2005.106

[48] J. Woodring, J. Ahrens, J. Figg, J. Wendelberger, S. Habib, and K. Heitmann. In-situ sampling of a large-scale particle simulation for interactive visualization and analysis. *Computer Graphics Forum*, 30(3):1151–1160, 2011. doi: 10.1111/j.1467-8659.2011.01964.x

[49] M. Zwicker, H. Pfister, J. van Baar, and M. Gross. EWA volume splatting. In *IEEE Visualization*, pp. 29–538, 2001. doi: 10.1109/VISUAL.2001.964490