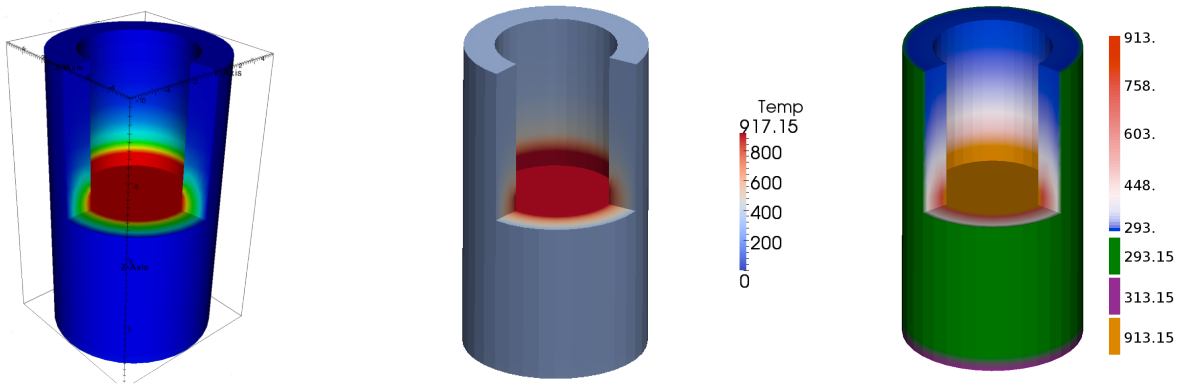
A Provably-Robust Sampling Method for Generating Colormaps of Large Data

David Thompson* Janine Bennett† C. Seshadhri‡ Ali Pinar§
Kitware, Inc. Sandia National Laboratories, Livermore, CA



(a) VisIt (b) ParaView (c) Prominent values

Figure 1: Temperature inside a rotating disk reactor. Sampling is able to identify constant boundary conditions and highlight them where they would otherwise be obscured by the full scalar value range.

Abstract

First impressions from initial renderings of data are crucial for directing further exploration and analysis. In most visualization systems, default colormaps are generated by simply linearly interpolating color in some space based on a value's placement between the minimum and maximum taken on by the dataset. We design a simple sampling-based method for generating colormaps that highlights important features. We use random sampling to determine the distribution of values observed in the data. The sample size required is independent of the dataset size and only depends on certain accuracy parameters. This leads to a computationally cheap and robust algorithm for colormap generation. Our approach (1) uses perceptual color distance to produce palettes from color curves, (2) allows the user to either emphasize or de-emphasize prominent values in the data, (3) uses quantiles to map distinct colors to values based on their frequency in the dataset, and (4) supports the highlighting of either inter- or intra-mode variations in the data.

Keywords: Color map, robust sampling, sublinear algorithm, CDF approximation.

Index Terms: G.3 [Probability and Statistics]: Probabilistic algorithms (including Monte Carlo) G.4 [Mathematical Software]: Parallel and vector implementations, Reliability and robustness, User interfaces H.5.2 [Information Interfaces and Presentation]: User Interfaces—Screen design (e.g., text, graphics, color)

*e-mail:david.thompson@kitware.com
†e-mail:jcbenne@sandia.gov
‡e-mail:scomand@sandia.gov
§e-mail:apinar@sandia.gov

1 Introduction

> Color is the most relative medium in art.
> — Josef Albers, *Interaction of Color*

Color is one of the most prevalent tools used in scientific visualization and possibly the easiest to misuse – knowingly, or, as this paper considers, unknowingly. As Albers notes, and cognitive neuroscience has established experimentally, human perception of color is only relative to its surroundings. Within a single image, color can be used to identify spatial trends at varying scales, discriminate between neighboring values, and even gauge absolute values to some degree.

However, there are very few tools to design the maps between numbers we wish to illustrate and colors that will aid in their undistorted perception[1]. General-purpose visualization tools are often given data with no description of its source, no units of measurement, no accuracy of its computation or measurement, no measurements of its trends, nor any indication of how importance is defined. Large datasets that cannot be held in memory (or require a high-performance computer with distributed memory) may not provide easy access to such summary information. It is difficult to choose a good colormap with some expert knowledge of the dataset.

Most default colormaps are defined by linearly interpolating dataset values between the maximum and minimum in the range of the dataset. This is computationally cheap but disregards the distribution of values within the data. Consider Figures 1(a) and 1(b), which show temperatures inside a rotating disk reactor. These were generated by the standard tools VisIt and ParaView respectively. It is not all clear from these figures that there are three special boundary conditions in this data. For example, the entire outer surface of the cylinder has the same temperature of 293.15°C, distinct from

[1]It is arguably impossible to say that a particular person's perception is biased or unbiased, but it may be possible to quantify how a population's perception of a particular feature is proportional to the evidence for it provided by the data *relative to other features in the same data*.

all other points. The bottom part has a temperature of 303.15°C, also distinct from all other points. These are completely obscured by the simplistic linear interpolation of color.

Clearly, choosing an informative colormap requires some data analysis. But analysis techniques providing detailed information about the data require possibly expensive pre-processing stages. For instance, information regarding relative frequencies of values and spatial relationships could be useful for a good colormap. But how does one obtain this information for a large dataset without affecting the running time of the actual visual presentation? In this work, we address this problem with an efficient method to generate colormaps via random sampling. These colormaps identify discrete values that occur with high probability – some fraction τ of the dataset or larger – and apply a continuous color-palette curve to the remaining CDF by quantile, so that perceptible changes in color are assigned to equiprobable ranges of values.

Theoretically, our algorithm is simple and provably robust. In practice, with a negligible overhead our algorithm yields images that better highlight features within the data. Most importantly, the required sample size depends only on desired accuracy parameters and is independent of the dataset size. However, as with any sampling approach, this technique may fail to discover exceedingly rare events; we relate sample size to the probability $1-\delta$ that an event more frequent than τ goes undetected. Events less frequent than τ are considered negligible. For reasonably small values of τ and δ, the sample size is practical – but it does grow quickly as τ is decreased.

Results

Consider visualizing a large dataset. If there are certain "prominent" values that occur with high frequency, then we might want these to take on special colors to highlight them. Continuous data may not uniformly occupy its range, and for some applications it is important to visualize overall trends in the colored attribute while in others it is small deviations from the trend that are important. We devise a new simple and scalable algorithm to design such color maps. The salient features of our algorithm follow.

- We introduce a simple sampling-based routine for approximately identifying prominent values and ranges in data. We provide a formal proof of correctness (including quantifiable error bounds) for this routine using probability concentration inequalities.
- The number of samples required by our routine only depends on the accuracy desired and *not on the data size*. For example, suppose we wish to find all values that occur with more than 1% frequency (in the data). Then, the number of samples required is some fixed constant, regardless of data size.
- Given these approximate prominent values and ranges in the data, we provide an automated technique for generating discrete and/or continuous color maps.

We empirically demonstrate our results on a variety of datasets. Consider Figure 1. The rightmost figure shows the coloring output by our algorithm, and notice how it picks up the three prominent values (and one range) in the data. This allows for a coloring that highlights the boundary conditions, as opposed to the standard colormaps.

2 Related Work

The most well-known work on color in scientific visualization is Brewer's treatise on the selection of color palettes [5–7]. Her thesis and much surrounding literature [13,21] focus on choosing palettes that 1) avoid confounding intensity, lightness, or saturation with hue either by co-varying them or using them to convey separate information; and 2) relate perceptual progressions of color with progressions of values in the data to be illustrated or – when the values being illustrated have no relationship to each other – to avoid perceptual progressions of colors so that the image does not imply a trend that is absent in the data. Other work [4, 18] discusses fundamental flaws with the commonly used default "rainbow" colormap and presents results on diverging color maps which have since been adopted by many in the community as they perform much better than the rainbow colormap in scientific visualization settings. Eisemann et al. [10] propose a family of pre-color-map transforms that transition from linear scaling (where all values in a range have the same importance) to a kind of histogram-equalized scaling (where values that frequently occur in the dataset are given more importance). They claim that linear scaling provides a way to discover outliers, but this is only true if the data has a single central tendency; outliers between multiple tendencies could well be masked by a linear scale. Also, perceptual differences between colors mapped to values are not considered. Finally the technique is not inherently scalable since it requires sorting all observed values, although it could likely be adapted to use representative subsamples. However, they identify a key factor in colormap design for exploratory visualization: without problem-specific knowledge, attempts to improve discrimination between values oppose attempts to remove unused ranges of values.

Other significant work has studied the generation of transfer functions for volume rendering. Here, work includes the use of entropy to maximize the "surprisal" and thus the information content of the image [2].The work of [14, 20] compare a number of transfer function generation techniques and provide good high-level overviews of the research in this area.

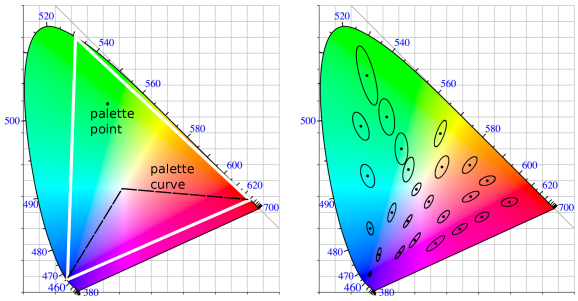
Borkin et al. [3] note that when a specific setting is being targeted, visualizations – including the colormap – should be chosen to match. However, in this paper we consider the task given to general-purpose visualization tools: how should default renderings of datasets provided without any context be created?

Finally, tone mapping [15] has been used to adjust images that contain more contrast than their presentation medium is able to provide by modeling how the human visual system deals with contrast.

3 Background

A *color model* is a mathematical abstraction in which colors are represented as tuples of values. Common color models include RGB, CMYK, and HSV. These and other color models differ in how the tuples of values are interpreted and combined to achieve the spectrum of attainable color values. For example, RGB uses additive color mixing, storing values of red, green, and blue. Not all devices can represent and capture colors equally. A *color space* defines a relationship between coordinates defined by a color model and the human perception of those coordinates – over some subset of coordinates that may be perceived. The *gamut* of a device is defined to be the subset of the color space that can be represented by the device, and those colors that cannot be expressed within a color model are considered out of gamut. CIELAB is a color space that was created to serve as a device-independent model to be used as a reference. It describes all colors that the human eye can see and is defined by three coordinates: *L* represents the lightness of a color, *a* represents its position between green and magenta/red, and *b* represents its position between yellow and blue.

The distance between colors is often defined in terms of the Euclidean distance between two colors in CIELAB space and is typically referred to as ΔE. Different studies have proposed different ΔE values that have a *just noticeable difference (JND)*. Often a value of $\Delta E \approx 2.3$ is used, however, several variants on the ΔE function have been introduced to address perceptual non-uniformities in the CIELAB color space. These non-uniformities can be visually depicted by *MacAdam ellipses*, which are elliptical regions that contain colors that are considered indistinguishable. These ellipses were identified empirically by MacAdam [16] who found that the size and orientation of ellipses vary widely depending on the test color. Figures 2(a) and 2(b) demonstrate the *chromaticities* (the quality of a color independent of its brightness), visible to the average human. In Figure 2(a) the white triangle is the gamut of the RGB color space and a *palette curve* and *palette point* are shown in



(a) (b)

Figure 2: The horse-shoe shape in these figures demonstrates the chromaticities visible to the average human. In (a) the white triangle is the gamut of the RGB color space and a palette curve and palette point are shown in this space. In (b) multiple MacAdam ellipses are shown to highlight perceptual non-uniformities in the color space.

this space. In Figure 2(b) multiple MacAdam ellipses are shown to highlight perceptual non-uniformities in the color space.

4 Overview

It is convenient to think of a dataset as a distribution $\mathscr{D}$ of values. Formally, pick a uniform random point in the data, and output the value at that point. This induces the distribution $\mathscr{D}$ on values we focus upon. We will fix $\tau \in (0,1)$ and positive integer ν as parameters to our procedure. We will set $\delta \in (0,1)$ as a failure probability for our algorithm.

We begin with identifying *prominent values*. These are values that make up at least a τ-fraction of the complete dataset. (In terms of $\mathscr{D}$, these are values with at least τ probability.) Once these are identified, these can be "removed" from $\mathscr{D}$ to get a distribution $\mathscr{D}'$. This can be viewed as modeling $\mathscr{D}$ with a mixture of a discrete distribution and another distribution $\mathscr{D}'$ that we approximate as continuous. Formally, $\mathscr{D}'$ is the distribution induced by $\mathscr{D}$ on all values except the prominent ones.

Next we divide the real line into ν intervals, where each interval has $1/\nu$ probability in $\mathscr{D}'$. This splitting provides an approximate CDF – to within $1/\nu$ with probability $1-\delta$ – that can be used for coloring data other than prominent values by quantile (i.e., histogram-equalized).

One of our contributions is a simple algorithm that (provably) approximately computes this in time that only depends on τ, ν, δ. It does not depend on the size of the actual data, and only on the required precision (which is quantified by these parameters).

To generate our colormap, we first assign the prominent values perceptually distant palette points. Intervals of equal probability are then distributed evenly along a palette curve parameterized by perceptual uniformity in order to make the distribution of data over the scale as perceptible as possible. This gives the final colormap. Finally, we provide a second colormap that alternates luminance between light and dark values for each equiprobable portion of $\mathscr{D}'$ in order to aid in identifying small local deviations within a single tendency of the data.

The interval identification is thus effectively a sample-based approximation to histogram equalization, however our algorithm performs this step *after* the prominent values have been separated from the dataset samples so that discrete behavior will not bias the density estimate [2].

[2] One example of mixed discrete and continuous behavior is rainfall totals; samples are generally modeled [23] as a mixture [11] of clear days (a discrete distribution with only 1 possible rainfall total) and rainy days (which tend to have an exponential or, more generally, gamma distribution of precipitation totals [12]). One might expect – and this paper demonstrates – similar behavior from simulations, where some regions are static due to boundary or initial conditions while others evolve into approximations of continuous behavior.

Because quantiles are less sensitive to outliers than the PDF, extreme values do not have an exaggerated effect on coloring. Our approach does not provide outlier identification and we contend that – for large data – outliers should be considered in a separate sampling biased toward them once prominent intervals have been confirmed as conceptually significant by a domain expert. Otherwise small samples are unlikely to be effective at identifying outliers.

It is important to note that sampling is important in order to obtain scalability [19]; when data is distributed across multiple processes with no guarantees on the uniformity of the distribution. A naive exact CDF computation can easily exceed the memory available to a single process, while bucketing can require significant memory and network bandwidth to properly compute.

The next section details the algorithm we use for computing colormaps; after that, we present the mathematics, both proof and algorithms, for sampling and for detecting prominent values and important intervals.

5 Mapping Scalar Values to Color

We formally describe our procedure for mapping data values to color according to the distribution of scalar values within the data. We assume the following are provided as input to the mapping algorithm: 1) a palette curve and palette points; and 2) a list of prominent values in the data and an approximate CDF for the remaining data. The CDF comprises a collection of n disjoint and contiguous intervals, $B_1, \ldots, B_n$. Each B_i has an associated minimum function value f_i^{min} and maximum function value f_i^{max}, simply corresponding to the left and right endpoints of B_i. Furthermore, each B_i also has an associated set of samples, whose size is denoted by $s(B_i)$. See the bottom of Figure 3(b), where the relative heights of bars in B_i denote the size of $s(B_i)$. These represent an approximate CDF in that (roughly speaking), the CDF value at the right endpoint of B_i is given by $\sum_{j=1}^{i} s(B_j) / \sum_{j=1}^{n} s(B_j)$. (In Section 6, we describe a provably robust sampling-based approach for the quick estimation of this information.)

Given the input, we first assign a unique color to each prominent value. Next, we discretize the palette curve p into k individual palette points of $\Delta E \approx 2.3$ (this value is tunable), see Figure 3(b).

Given a scalar value, f we compute an interpolation factor, t_f, based on the position of f in the CDF. (Again, refer to Figure 3(b).) To do this we identify the bucket B_i that contains f and compute:

$$t_f = \frac{\sum_{j=1}^{i-1} s(B_j) + \left(\frac{f - f_i^{min}}{f_i^{max} - f_i^{min}}\right) * s(B_i)}{\sum_{j=1}^{n} s(B_j)}$$

Once we have identified t_f, we compute the final color, c_f, using the palette curve p with k palette points as

$$j = t_f * k,$$
$$t_j = j - \texttt{floor}(j),$$
$$c_f = c_j + t_j(c_{j+1} - c_j).$$

Implementation details: As Eisemann et al. note [10], colormap definitions have opposing objectives in exploratory visualization where problem specifics are unknown; it is impossible to discern whether similar values should be perceptually similar in order that trends across large differences in function values may be detected or whether similar values should be perceptually distinct in order that small differences may be detected. We provide two colormaps for these two situations that users must choose between: an *inter-mode* map that varies hues smoothly and at constant luminance within CDF intervals and an *intra-mode* colormap that varies hue

smoothly but rapidly alternates luminance between high and low values to provide visual cues for discriminating between small differences in value. We name the situations inter- and intra-mode because one might find the former useful for contrasting behavior across different spatial or statistical modes; and the latter useful for contrasting behavior within a given mode. This use of luminance is similar to the use of structured light to highlight small geometric features [24]. Furthermore, prominent values can be emphasized or de-emphasized within an image by modifying the lightness of the associated color in CIELAB space. We use the Little CMS color engine [17] to perform all transformations between color spaces and compute ΔE distances between colors.

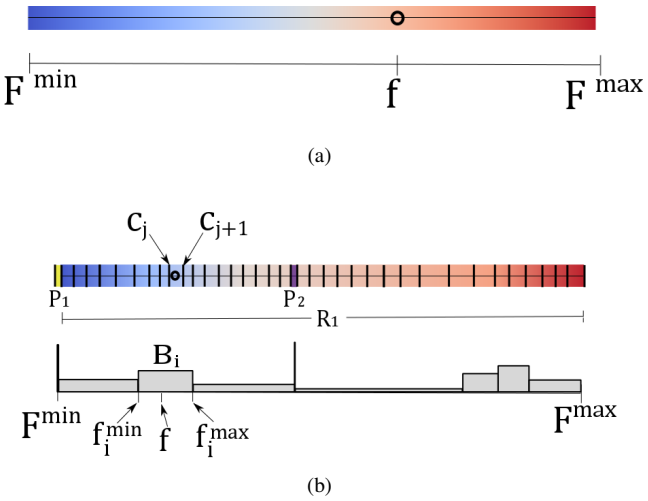


Figure 3: Consider a dataset with two prominent values, P_1, P_2 and overall range R_1. In figure (a) a simple linear interpolation scheme is depicted in which the color associated with the value f is independent of the distribution of values in the dataset. In (b) the two prominent values are assigned the colors yellow and purple respectively. The range is assigned blue-red. Using our CDF-based interpolation scheme, we obtain a color for the function value f that is determined by the important interval and associated range to which f belongs. In out future work, we point out that should we have a way to split ranges, a second green-orange palette curves might be used to illustrate samples from apparently distinct populations.

6 Prominent Values and the CDF Approximation

We describe the sampling approaches used to determine prominent values and important intervals of a dataset. We employ two simple sampling algorithms for this purpose. The first algorithm does a direct sampling to determine frequent values in the dataset. The second algorithm constructs a series of intervals, such that the frequency of data within each interval is roughly the same. Note that a large variance in the lengths of these intervals indicates a non-uniformity in data value distribution. These intervals represent our approximate CDF.

We treat our data as a *discrete* distribution, where for each value r, p_r is the fraction of the dataset where the value r is attained (So $\{p_r\}$ describes a distribution over the range of the dataset). We use $\mathscr{D}$ to denote this distribution and R to denote the support. For any set S (often an interval of the real line), we use $P(S)$ to denote the probability mass of S.

The analysis of both algorithms follow from straightforward applications of Chernoff bounds. We state the *multiplicative Chernoff bound* (refer to Theorem 1.1 in [9]) for sums of independent random variables.

Theorem 6.1. *[Chernoff bound] Let $X_1, X_2, \ldots, X_k$ be independent random variables in $[0,1]$ and $X = \sum_{i=1}^{k} X_i$.*

- *(Lower tail) For any $\varepsilon > 0$,*

$$\Pr[X < (1-\varepsilon)\mathbf{E}[X]] \leq \exp(-\varepsilon^2 \mathbf{E}[X]/2).$$

- *(Upper tail) For any $\varepsilon > 0$,*

$$\Pr[X > (1+\varepsilon)\mathbf{E}[X]] \leq \exp(-\varepsilon^2 \mathbf{E}[X]/3).$$

- *(Upper tail) For any $t > 2e\mathbf{E}[X]$,*

$$\Pr[X > t] \leq 2^{-t}.$$

In our theorems, we do not attempt to optimize constants. The running time of our algorithms does not depend on the data size, and the theorems are basically proof of concepts. Our empirical work will show that the actual samples sizes required are quite small. For convenience, we use c and c' to denote sufficiently large constants. Our algorithms will take as input a sample size parameter. Our theorems will show that this can be set to a number that only depends on precision parameters, for desired guarantees.

6.1 Finding prominent values

Our aim is to determine values of r such that $p_r > \tau$ is large, where $\tau \in (0,1)$ is a *threshold parameter*.

`find-prominent`(s, τ)
Inputs: sample size s, threshold τ
Output: the "frequent set" of range elements, I.

1. Generate set S of s independent random samples from $\mathscr{D}$
2. Initialize important set $I = \emptyset$.
3. For any element $r \in \mathscr{D}$ that occurs more than $s\tau/2$ times in S,
   Add r to I.
4. Output I.

The following theorem states that (up to some approximation), I is indeed the set of frequent elements. The constants in the following are mainly chosen for presentation. (Instead of $p_r < \tau/8$ in the following, we can set it to τ/α, for any $\alpha > 1$, and choose s accordingly.) Throughout our theorems, we use δ for a tunable error parameter that decides the sample size s. Note that it is not an explicit parameter to the algorithms.

Theorem 6.2. *Set $s = (c/\tau)\ln(c/(\tau\delta))$. With probability $> 1-\delta$ (over the samples), the output of* `find-prominent`(τ, δ) *satisfies the following.*

If $p_r > \tau$, then $r \in I$.
If $p_r < \tau/8$, then $r \notin I$.

Proof. We first prove that with probability at least $1-\delta/2$, for all $p_r > \tau$, $r \in I$. Then we show that with probability at least $1-\delta/2$, for all $p_r < \tau/8$, $r \notin I$. A union bound then completes the proof.

Consider some r such that $p_r > \tau$. Let X_i be the indicator random variable for the event that the ith sample is r. So $\mathbf{E}[X_i] = p_r$ and all X_is are independent. We set $X = \sum_{i \leq s} X_i$ and apply the Chernoff lower tail of Theorem 6.1 with $\varepsilon = 1/2$. Hence, $\Pr[X < \mathbf{E}[X]/2] \leq \exp(-\mathbf{E}[X]/8)$. Note that $\mathbf{E}[X] = p_r s > s\tau$. We obtain $\Pr[X < s\tau/2] \leq \Pr[X < \mathbf{E}[X]/2] \leq \exp(-\mathbf{E}[X]/8) \leq \exp(-s\tau/8)$. Since $s = (c/\tau)\ln(c/(\tau\delta))$, $\exp(-s\tau/8) = \exp(-c\ln(c/(\tau\delta)) \leq \delta\tau/16$.

Putting it together, $\Pr[X < s\tau/2] < \delta\tau/16$. Hence, in our algorithm, the element r will *not* be in I with probability at most $\delta\tau/16$. There are at most $1/\tau$ values of r such that $p_r > \tau$. By the union bound, the probability that there exists some such value of r occurring less than $s\tau/2$ times is at most $\delta/16$. Hence, with probability $> 1-\delta/16$, $\forall p_r > \tau$, $r \in I$.

Define set $A = \{r | p_r \geq \tau/8\}$, and $R' = R \setminus A$. The second part is stated as Lemma 6.3 below with $\alpha = \tau/8$. We get that with probability $> 1-\delta/2$, all $r \in R'$ individually occur less than $s\tau/2$ times. Hence, none of them are in I. □

Lemma 6.3. *Let $\alpha \in (0,1)$ and $s > (c/8\alpha)\ln(c/(8\alpha\delta))$. Consider a set R' such that $\forall r \in R'$, $0 < p_r \le \alpha$. With probability $> 1-\delta/2$, the following holds. For all $r \in R'$, the number of occurrences of r in s uniform random samples from $\mathscr{D}$ is at most $4s\alpha$.*

Proof. We apply Claim 6.4 (given below). This gives a series of intervals $R_1, R_2, \ldots, R_n$, such that for all $m < n$, $P(R_m \cap R') \in [\alpha, 2\alpha]$. Also, $P(R_n \cap R') \le 2\alpha$.

Consider some R_m for $m < n$, and let Y_i be the indicator random variable for the ith sample falling in R_m. We have $\mathbf{E}[Y_i] \in [\alpha, 2\alpha]$ and $\mathbf{E}[Y] \in [\alpha s, 2\alpha s]$ (where $Y = \sum_{i \le s} Y_i$). It will be convenient to use the bound $\mathbf{E}[Y_i] \in [\alpha/2, 2\alpha]$ and $\mathbf{E}[Y] \in [\alpha/2s, 2\alpha s]$.

Since the Y_is are independent, we can apply the first Chernoff upper tail with $\varepsilon = 1$. This yields $\Pr[Y > 2\mathbf{E}[Y]] \le \exp(-\mathbf{E}[Y]/3)$. Since $\mathbf{E}[Y] \le 2\alpha s$, $\Pr[Y > 4s\alpha] \le \Pr[Y > 2\mathbf{E}[Y]]$. Since $\mathbf{E}[Y] \ge \alpha s/2$, $\exp(-\mathbf{E}[Y]/3) \le \exp(-\alpha s/6) = \delta\alpha/3$. Putting it together, $\Pr[Y > 4s\alpha] < \delta\alpha/3$. Hence, $\Pr[Y > 4s\alpha] \le \exp(-\alpha s/3) = \delta\alpha/3$.

Now focus on R_n and define Y analogous to above. If $P(R_n \cap R') > \alpha/2$, we can apply the previous argument with $\varepsilon = 1$. Again, we deduce that $\Pr[Y > 4s\alpha] \le \exp(-s\alpha/6) = \delta\alpha/3$. If $P(R_n \cap R') < \alpha/2$, we apply the second Chernoff tail with $t = 4s\alpha$ (observing that $4s\alpha \ge (2e)\alpha/2$) to deduce that $\Pr[Y > 4s\alpha] \le 2^{-4s\alpha} \le \delta\alpha/3$.

We apply the union bound over all R_m for $m \le n$. Note that n is at most $1/\alpha + 1$, since $P(R_m \cap R') \ge 2\alpha$ and the R_ms are disjoint. So with probability at most $\delta/2$, there exists some R_d such that number of occurrences in R_d is more than $4s\alpha$. Hence, with probability at least $1-\delta/2$, no element in R' can appear more than $4s\alpha$ times. □

Claim 6.4. *Let $\alpha \in (0,1)$. Consider a set R' such that $\forall r \in R'$, $0 < p_r \le \alpha$. There exists a sequence of numbers $\min_{r \in R'} r = z_1, z_2, \ldots, z_k = \max_{r \in R'} r$ ($k \ge 2$) such that for all $i < k-1$, $P([z_i, z_{i+1}) \cap R') \in [\alpha, 2\alpha]$ and $P([z_{k-1}, z_k]) \le 2\alpha$.*

Proof. This is done through a simple iterative procedure. We start with $z_1 = \min_{r \in R'} r$. Given z_i, we describe how to find z_{i+1}. Imagine z_{i+1} initialized to z_i and continuously increased until the $P([z_i, z_{i+1}] \cap R')$ (note that we use a closed interval) exceeds 2α. If z_{i+1} crosses $\max_{r \in R'} r$, then we have found the last interval and terminate this process. Now, $P([z_i, z_{i+1}) \cap R')$ (the open interval) must be less than 2α, or we would have stopped earlier. Furthermore, $P([z_i, z_{i+1}) \cap R') = P([z_i, z_{i+1}] \cap R') - p_{z_{i+1}} \ge 2\alpha - \alpha = \alpha$. □

6.2 Finding an approximate CDF

Our aim is to construct a series of disjoint intervals that (almost) equally partition the probability mass. To gain some intuition, consider a positive integer v and a sequence of numbers $y_0, y_1, \ldots, y_v$ where $y_0 = \min_{r \in R} r$, $y_v = \max_{r \in R} r$, and for all $i < v$, $P([y_i, y_{i+1})) = 1/v$. Our algorithm will try to find these intervals (for a parameter v). Of course, such intervals may not even exist, due to the discrete nature of $\mathscr{D}$. Nonetheless, we will try to find suitable approximations. We assume that there are no values in $\mathscr{D}$ with high probability. This is an acceptable assumption, since we run this procedure after "removing" prominent values from $\mathscr{D}$.

There are two parameters for `find-CDF`: the sample size s and the block size b. For convenience, assume b divides s. The output is a series of s/b intervals, each with (provably) approximately the same probability mass. As we mentioned earlier, this constitutes an approximation to the CDF, since the probability mass of the first k of these intervals will be (approximately) proportional to k.

`find-CDF`(s, b)
Inputs: sample size s, block size b
Outputs: Intervals $B_1, B_2, \ldots$

1. Generate set S of s independent random samples from $\mathscr{D}$.
2. Sort these to get the (ordered) list $\{x_1, x_2, x_3, \ldots, x_s\}$.
3. Output the intervals $B_1 = [x_1, x_b)$, $B_2 = [x_{b+1}, x_{2b})$, etc. In general, the ith interval B_i is $[x_{(i-1)b+1}, x_{ib})$ and there are s/b blocks. The samples in this interval form the associated set, so $s(B_i) = |B_i \cap S|$.

This main theorem involves some play of parameters, and we express s and b in terms of an auxiliary (integer) parameter v. Again, the constants chosen here are mainly given for some concreteness and notational convenience. We use the notation $A \in (1 \pm \beta)B$ as a shorthand for $A \in [(1-\beta)B, (1+\beta)B]$.

Theorem 6.5. *Set $s = cv\ln(cv/\delta)$ and $b = s/v$. Suppose there exists no $r \in R$ such that $p_r > 1/100v$. With probability $> 1-\delta$, the following holds. For each output interval B, $P(B) \in (1 \pm 1/10)/v$. Furthermore, $P((\min_{r \in R} r, x_1))$ and $P((x_s, \max_{r \in R} r))$ are at most $1/50v$.*

Proof. We first apply Claim 6.4 with $\alpha = 1/100v$ and $R' = R$, to get the sequence $\min_{r \in R} r = z_1, z_2, \ldots, z_k = \max_{r \in R} r$. We have $P([z_i, z_{i+1})) \in [1/100v, 1/50v]$ for $i < k-1$ and $P([z_{k-1}, z_k]) \le 1/50v$. We prove the following lemma.

Lemma 6.6. *Set $s = cv\ln(cv/\delta)$. Let $Z_{i,j}$ be the number of samples falling in interval $[z_i, z_j)$. With probability at least $1-\delta/2$, for all pairs i, j, $Z_{i,j} \in (1 \pm 1/20)sP([z_i, z_j))$.*

Proof. Fix interval $[z_i, z_j)$. Let X_m be the probability of the mth sample falling in $[z_i, z_j)$. We have $\mathbf{E}[X_m] = P([z_i, z_j)) \ge 1/100v$ and all X_m's are independent. Also, $Z_{i,j} = \sum_{m \le s} X_m$, and $\mathbf{E}[Z_{i,j}] = sP([z_i, z_j)) \ge s/100v$. The upper Chernoff tail with $\varepsilon = 1/20$ yields $\Pr[Z_{i,j} < (1-1/20)sP([z_i, z_j))] \le \exp(-sP([z_i, z_j))/800)$. The lower tail with $\varepsilon = 1/20$ yields $\Pr[Z_{i,j} > (1+1/20)sP([z_i, z_j))] \le \exp(-sP([z_i, z_j))/1200)$.

By a union bound over both tails and plugging in the bound $P([z_i, z_j)) \ge 1/100v$, the probability that $X_{i,j} \notin (1 \pm 1/20)sP([z_i, z_j))$ is $\exp(-s/(800 \cdot 100v)) + \exp(-s/(1200 \cdot 100v))$. Doing the calculations (for sufficiently large constant c), this probability is at most δ/cv^2. A union bound over all intervals (at most $\binom{100v+1}{2}$ of them) completes the proof. □

We apply Lemma 6.6, so with probability $1-\delta/2$, for all i, j, $Z_{i,j} \in (1 \pm 1/20)sP([z_i, z_j])$. Now, we apply Lemma 6.3 with $\alpha = 1/100v$ and $R' = R$. With probability $> 1-\delta/2$, no element occurs more than $s/25v$ times. By a union bound, both these hold with probability $> 1-\delta$. Under these conditions, we complete our proof.

Fix a block B, and let $[z_i, z_j)$ be the smallest such interval that contains B. Let $[z_{i'}, z_{j'})$ be the largest such interval inside B. Since (for any a) $P([z_a, z_{a+1})) \le 1/50v$, $P([z_i, z_j)) - 2 \cdot (1/50v) \le P(B) \le P([z_i, z_j))$. Similarly, $P([z_{i'}, z_{j'})) \le P(B) \le P([z_{i'}, z_{j'})) + 2 \cdot (1/50v)$.

Let ℓ denote the number of sample points in $B = [x_h, x_{h+1})$. Since the number of samples in $[x_h, x_{h+1}]$ (the closed interval) is exactly b, $\ell \le b = s/v$. Also, ℓ is at least b minus the number of sample occurrences of x_{h+1}. So $\ell \ge b - s/25v = (1-1/25)s/v$. We remind the reader that $Z_{i,j}$ is the number of sample occurrences in $[z_i, z_j)$, which is an interval containing B. Hence $\ell \le Z_{i,j}$. Similarly, $\ell \ge Z_{i',j'}$.

We use the bound that $Z_{i,j} \in (1 \pm 1/20)sP([z_i, z_j))$ (similarly for $Z_{i',j'}$). Since $\ell \le X_{i,j}$, we get $(1-1/25)s/v \le (1+1/20)sP([z_i, z_j))$. Since $\ell \ge X_{i',j'}$, $s/v \ge (1-1/20)sP([z_{i'}, z_{j'})$. Relating the probabilities to $P(B)$, we get $(1-1/25)/v \le (1+1/20)(P(B) + 1/25v)$ and $1/v \ge (1-1/20)(P(B) - 1/25v)$. Rearranging the terms, we complete the proof for output interval B.

No samples fall in the intervals $(\min_{r \in R} r, x_1)$ and $(x_s, \max_{r \in R} r)$. Hence, they must be completely contained in some interval of the form $[z_a, z_{a+1}]$, and their probabilities are at most $1/50v$.

□

7 Results

In this section we demonstrate the results of applying our colormap generation algorithm to a variety of datasets. These include the Mandelbrot dataset, a rotating disk reactor, and two combustion

datasets. We compare images of the data created with default colormaps of several visualization tools to colormaps created using our technique.

The Mandelbrot dataset is a synthetic function that can be sampled at any resolution. By contouring the volumetric function using a geometric sequence of isovalues, we obtained auxiliary data used to characterize how prominent value detection behaves as distinct values become near enough to appear as a continuous range, see Figure 6. We also run the algorithm on the data without contouring in Figure 7.

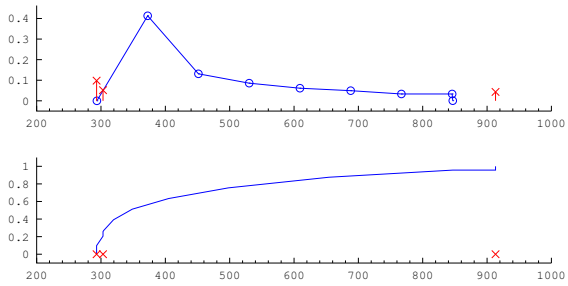


Figure 4: The probability density function (PDF, top) and cumulative density function (CDF, bottom) of the rotating disk reactor, shown in blue and obtained from the prominent values and blocked ranges. Prominent values and their probabilities are shown in red. Note that prominent values have an absolute probability associated with them while the PDF shows densities estimated from many different values; none of the samples used to estimate the density occur with frequencies approaching that of the prominent values.

The rotating disk reactor, shown in Figure 1, is a steady-state simulation of continuous vapor deposition (CVD) carried out by MPSalsa [22, § D.2]. A notch has been removed for illustrative purposes. It is a small dataset that contains thermodynamic state variables, velocity, and chemical species concentrations at each node. The simulated region is a fluid-filled cavity between a concentric tube and rod where reactants flow up from the unobstructed volume (bottom), across the heated rod, and exit at the annulus (top). Of interest is the fact that temperature boundary conditions have been imposed: the outer cylindrical wall is held at an ambient temperature of 293.15 K, the inner rod cap is heated to 913.15 K, and the fluid entering the chamber is a constant 303.15 K. Because the two lower temperature conditions are very near each other relative to the heated rod cap, it is impossible to distinguish them in the default views of ParaView and VisIt. However, by passing the temperature through the sampling algorithm above, we obtain the PDF and CDF shown in Figure 4, along with associated prominent values that pull out these features. By assigning colors far from the palette curve to these prominent values, the difference is apparent.

In addition to these two datasets we also demonstrate our results on two combustion datasets: HCCI, and Lifted Ethylene Jet. These datasets were generated by S3D [8], a turbulent combustion simulation code that performs first principles-based direct numerical simulations in which both turbulence and chemical kinetics introduce spatial and temporal scales spanning typically at least 5 decades. The HCCI dataset is a study of turbulent auto-ignitive mixture of Di-Methyl-Ether and air under typical Homogeneous Charge Compression Ignition (HCCI) conditions [1]. This simulation is aimed at understanding the ignition characteristics of typical bio-fuels for automotive applications and has a domain size of over 175 million grid points. The second simulation describes a lifted ethylene jet flame [25], involved in a reduced chemical mechanism for ethylene-air combustion, with a domain size of 1.3 billion grid points.

Figure 7 contains images comparing our technique with that of ParaView for the Mandelbrot, HCCI, and Lifted Ethylene Jet datasets. The first row demonstrates default color maps generated by ParaView for the Mandelbrot, HCCI, and Lifted Ethylene Jet

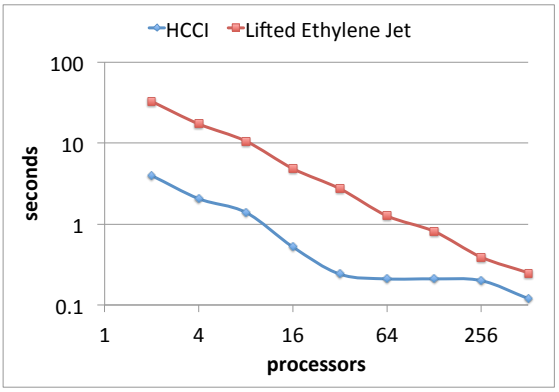


Figure 5: Our technique demonstrates good scalability as the number of samples required by our algorithm is fixed according to accuracy parameters as opposed to dataset size.

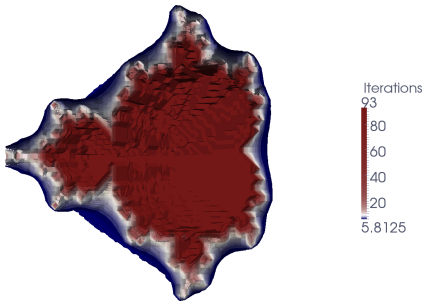
datasets. The second and third row contrast inter- vs. intra-mode differences with the prominent values emphasized as colors with low lightness in CIELAB space, while the fourth and fifth row contrast inter- vs. intra-mode differences with prominent values de-emphasized as colors with high lightness in CIELAB space. In particular, within the combustion datasets, our approach visually depicts structures in the data that are difficult to see in the default view. These structures cover a relatively small percentage of overall scalar domain, however these values are observed with relatively high frequency within the data. We note that our approach can be effective at identifying coherent structures in spite of the fact that we are not integrating spatial information into our estimates.

Scalability: Figure 5 highlights the scalability of our approach for the Lifted Ethylene Jet and HCCI datasets. We performed our experiments on Lens at the Oak Ridge Leadership Computing Facility. Lens is a 77-node commodity-type Linux cluster whose primary purpose is to provide a conduit for large-scale scientific discovery via data analysis and visualization of simulation data. Lens has 45 high-memory nodes that are configured with 4 2.3 GHz AMD Opteron processors and 128 GB of memory. The remaining 32 GPU nodes are configured with 4 2.3 GHz AMD Opteron processors and 64 GB of memory. Datasets were evaluated with $\tau = 0.001$ and $b = 1024$, requiring approximately 100,000 samples to achieve a failure probability of $\delta = 10^{-6}$.

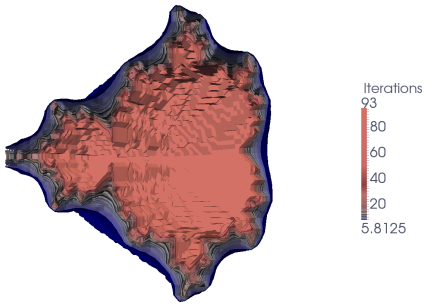
8 Conclusion

We introduced a sampling based method to generate colormaps. Our algorithm identifies important values in the data, provides a provably-good approximation of the CDF of the remaining data, and uses this information to automatically generate discrete and/or continuous color maps. Our experiments showed the new colormaps yield images that better highlight features within the data. The proposed approach is simple and efficient, yet provably robust. Most importantly, the number of samples required by the algorithm depends only on desired accuracy in estimations and is independent of the dataset size. This provides excellent scalability for our algorithms, as the required preprocessing time remains constant despite increasing data as we move to exascale computing. The algorithms are also efficiently parallelizable and and yield linear scaling.

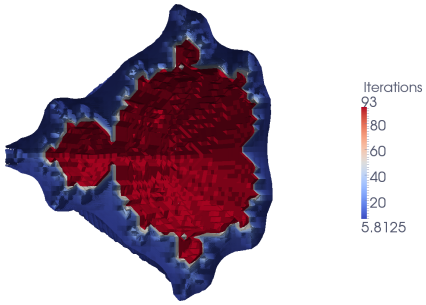
In our future work, we would like to identify regions of the CDF that are flat to within some tolerance (i.e., regions of the PDF that are nearly empty) in order to split the CDF into ranges likely to be generated by different distributions which have been mixed together by sampling. These ranges would each be colored with different palette curves in order to provide visual feedback about spatial regions that are likely sampling different distributions.



(a) Inter-mode



(b) Intra-mode



(c) ParaView

Figure 6: These images show 32 contour values, each of which is identified by our algorithm as a prominent value. Many of these isovalues lie closely together and are difficult to differentiate using traditional default color maps. Using our algorithm it is much easier to see that there are in fact many individual surfaces.

Acknowledgments

The authors wish to thank Dr. Jacqueline Chen for access to the combustion use case examples used in this paper. This work is supported by the Laboratory Directed Research and Development (LDRD) program of Sandia National Laboratories. Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

References

[1] G. Bansal. *Computational Studies of Autoignition and Combustion in Low Temperature Combustion Engine Environments*. PhD thesis, University of Michigan, 2009.

[2] U. Bordoloi and H.-W. Shen. View selection for volume rendering. In *Vis '05: Proceedings of the IEEE Visualization 2005*, pages 487–494, 2005.

[3] M. A. Borkin, K. Z. Gajos, A. Peters, D. Mitsouras, S. Melchionna, F. J. Rybicki, C. L. Feldman, and H. Pfister. Evaluation of artery visualizations for heart disease diagnosis. *IEEE Transactions on Visualization and Computer Graphics*, 17(12), 2011.

[4] D. Borland and R. M. T. II. Rainbow color map (still) considered harmful. *IEEE Computer Graphics & Applications*, pages 14–17, Mar. 2007.

[5] C. A. Brewer. Guidelines for selecting colors for diverging schemes on maps. *The Cartographic Journal*, 33(2):79–86, 1996.

[6] C. A. Brewer. A transition in improving maps: The ColorBrewer example. *Cartography and Geographic Information Science*, 30(2):159–162, 2003.

[7] C. A. Brewer, G. W. Hatchard, and M. A. Harrower. ColorBrewer in print: A catalog of color schemes for maps. *Cartography and Geographic Information Science*, 30(1):5–32, 2003.

[8] J. H. Chen, A. Choudhary, B. de Supinski, M. DeVries, E. R. Hawkes, S. Klasky, W. K. Liao, K. L. Ma, J. Mellor-Crummey, N. Podhorski, R. Sankaran, S. Shende, and C. S. Yoo. Terascale direct numerical simulations of turbulent combustion using s3d. *Computational Science and Discovery*, 2:1–31, 2009.

[9] D. Dubhashi and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

[10] M. Eisemann, G. Albuquerque, and M. Magnor. Data driven color mapping. In *Proceedings of EuroVA: International Workshop on Visual Analytics*, Bergen, Norway, May 2011.

[11] K. R. Gabriel and J. Neumann. A Markov chain model for daily rainfall occurrence at Tel Aviv. *Quarterly Journal of Royal Meteorology Society*, 88:90–95, 1962.

[12] N. T. Ison, A. M. Feyerherm, and L. Bark. Wet period precipitation and the gamma distribution. *Journal of Applied Meteorology*, 10:658–665, 1971.

[13] G. Kindlmann, E. Reinhard, and S. Creem. Face-based luminance matching for perceptual colormap generation. In *Proceedings of the IEEE Visualization*, 2002.

[14] 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.

[15] G. Larson, H. Rushmeier, and C. Piatko. A visibility matching tone reproduction operator for high dynamic range scenes. *IEEE Transactions on Visualization and Computer Graphics*, 3(4), Dec. 1997.

[16] D. L. MacAdam. "visual sensitivities to color differences in daylight". *JOSA*, 32(5):247–274, 1942.

[17] M. Maria. Little CMS: How to use the engine in your applications, 2012.

[18] K. Moreland. Diverging color maps for scientific visualization. In G. Bebis, R. D. Boyle, B. Parvin, D. Koracin, Y. Kuno, J. Wang, R. Pajarola, P. Lindstrom, A. Hinkenjann, M. L. Encarnação, C. T. Silva, and D. S. Coming, editors, *Advances in Visual Computing, 5th International Symposium, ISVC 2009, Las Vegas, NV, USA, November 30 - December 2, 2009, Proceedings, Part II*, volume 5876 of *Lecture Notes in Computer Science*, pages 92–103. Springer, 2009.

[19] P. Pébay, D. Thompson, and J. Bennett. Computing contingency statistics in parallel: Design trade-offs and limiting cases. In *Proceedings of the IEEE International Conference on Cluster Computing*, 2010.

[20] H. Pfister, W. E. Lorensen, W. J. Schroeder, C. L. Bajaj, and G. L. Kindlmann. The transfer function bake-off (panel session). In *IEEE Visualization*, pages 523–526, 2000.

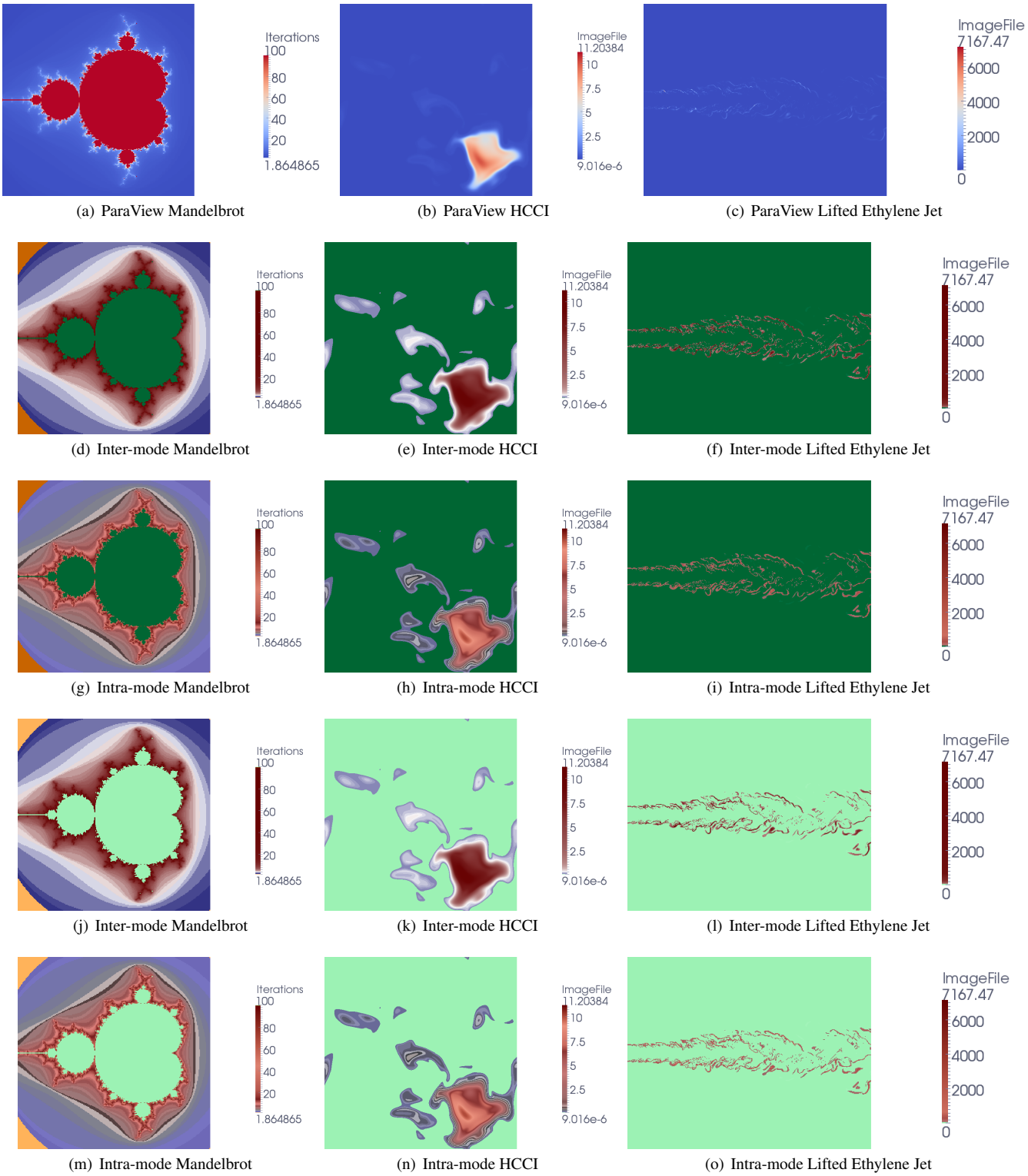
[21] B. E. Rogowitz, L. A. Treinish, and S. Bryson. How not to lie with visualization. *Computers in Physics*, 10(3):268–273, 1996.

[22] A. Salinger, K. Devine, G. Hennigan, H. Moffat, S. Hutchinson, and J. Shadid. MPSalsa: A finite element computer program for reacting flow problems, part 2 – users guide. Technical Report SAND96-2331, Sandia National Laboratories, Sept. 1996.

[23] J. Suhaila and A. A. Jemain. Fitting daily rainfall amount in peninsular malaysia using several types of exponential distributions. *Journal of Applied Sciences Research*, 3(10):1027–1036, 2007.

[24] P. M. Will and K. S. Pennington. Grid coding: a preprocessing technique for robot and machine vision. In *Proc. Int. Joint Conf. on Artificial Intelligence*, pages 66–70, 1971.

[25] C. S. Yoo, R. Sankaran, and J. H. Chen. Three-dimensional direct numerical simulation of a turbulent lifted hydrogen jet flame in heated coflow: Flame stabilization and structure. *Journal of Fluid Mechanics*, 640:453–481, 2009.



(a) ParaView Mandelbrot (b) ParaView HCCI (c) ParaView Lifted Ethylene Jet

(d) Inter-mode Mandelbrot (e) Inter-mode HCCI (f) Inter-mode Lifted Ethylene Jet

(g) Intra-mode Mandelbrot (h) Intra-mode HCCI (i) Intra-mode Lifted Ethylene Jet

(j) Inter-mode Mandelbrot (k) Inter-mode HCCI (l) Inter-mode Lifted Ethylene Jet

(m) Intra-mode Mandelbrot (n) Intra-mode HCCI (o) Intra-mode Lifted Ethylene Jet

Figure 7: The first row demonstrates default color maps generated by ParaView for the Mandelbrot, HCCI, and Lifted Ethylene Jet datasets. The second and third row contrast inter- vs. intra-mode differences with the prominent values emphasized as colors with low lightness in CIELAB space, while the fourth and fifth row contrast inter- vs. intra-mode differences with prominent values de-emphasized as colors with high lightness in CIELAB space.