A Feature-first Approach to Clustering for Highlighting Regions of Interest in Scientific Data

Robert Sisneros
National Center for Supercomputing Applications, Urbana, Illinois, U.S.A.
sisneros@illinois.edu

Abstract
We present a clustering algorithm that classifies the points of a dataset by a combination of scalar variables' values as well as spatial locations. How heavily the spatial locations impact the algorithm is a tunable parameter. With no impact the algorithm bins the data by calculating a histogram and classifies each point by a bin ID. With full impact, points are bunched together by spatial neighborhood regardless of value. This approach is unsurprisingly very sensitive to this weighting; a sampling of possible values yields a wide variety of classifications. However, we have found that when tuned just right it is indeed possible to extract meaningful features from the resulting clustering. Furthermore, the principles behind our development of this technique are also applicable in both tuning the algorithm as well as in selecting data regions. In this paper we will provide the details of design and implementation and demonstrate using the auto-tuned approach to extract interesting regions of real scientific data. Our target application is the automatic detection of land cover data anomalies in NASA’s Moderate Resolution Imaging Spectroradiometer (MODIS) sensors.

Keywords: Visualization, Anomaly Detection, Parallel Computing, MODIS

1 Introduction
Scientists rely on the analysis of simulation, observed, and derived data for hypothesis verification or discovery, and many tools and methods exist to help with either, or some combination of both. The aim of this work is to aid discovery via detection of features in scientific data, such as the identification of an area of land cover data sustaining prolonged effects of a natural disaster.

Finding salient features in scientific data is a demanding task. In addition to the domain expertise required to hypothesize potential intervariable interactions it is also necessary to incorporate the analytical expertise to numerically abstract such an interaction so as to be generally measurable. Furthermore, this is likely only the point of entry for a successful technique. Finding features is more than demanding; it is daunting. We posit that regardless of domain or how “salient features” are defined, finding them is hard. Our approach is rooted in assuming the converse is true as well. Specifically, we start with an obtuse definition of a feature as some collection of data locations; we call one that is hard to find a
salient feature. In this context, the result of any clustering algorithm is a set of features and a measure of how difficult one is to find is a measure of its importance. As such, our proposed algorithm is conceptually extremely simple. In some cases however this simplicity was dictated by taking this feature-first approach.

In this paper, we will present a simple clustering algorithm that bins data by a combination of value and location. We call this hybrid of histogram-style binning and spatial locations the histospam. Our analysis now shifts from answering domain specific questions about data to answering the following: if a feature is as defined above, and therefore some histospam bin, what could make it hard to find? Common sense tells us that things are hard to find when they are: (1) small, i.e. they are spread thinly throughout the space or only a few points; (2) few, i.e. they are outliers; (3) hard to see, i.e. they are buried in the data, next to or inside of larger clusters. In addressing the problem in this way we have the more manageable sounding task of finding a few, small, hard-to-see things.

It is with this philosophy that we make all choices in the histospam. We will select the weight between value and location in a way similar to deciding which values to threshold before viewing. We will demonstrate some particulars of the histospam using a well known, toy volumetric dataset before validating the method on our target application data: the Normalized Difference Vegetation Index from NASA's Moderate Resolution Imaging Spectroradiometer (MODIS) sensors [11]. The reduced dimensionality of the MODIS landscape data in principle suggests it should be much less likely to exhibit certain indicators of “hard to find”-ness, e.g. it would be much harder for a cluster to be “buried” in data. The histospam, with its spatial element, differs from the typical clustering approaches used to analyze this target data.

In the remainder of this paper, we first review some related work in Section 2. We will then provide details of the histospam algorithm in Section 3 and follow with a discussion on its simplicity in Section 4. Then, in Section 5 we detail results of clustering via the histospam and conclude in Section 6.

2 Background

The notion primary to our approach is that even as typical model-based clustering methods [5] are routinely used to classify [25] and visualize scientific data [35], the perception of distance in a visual setting is disjoint from that used in clustering. By blurring this line and refactoring methods based on this visual distinction, we are effectively refactoring foundational approaches to be more natively visualization-applicable. Furthermore, this visual technique is applicable in the detection of salient regions in multivariate scientific data [14].

Regarding an early feature detection problem, finding vortices, The Visualization Handbook [15] defines a feature as “a pattern occurring in a dataset that is the manifestation of correlations among various components of the data.” Correlations are readily calculable and serve as an excellent starting point for detecting features. Indeed, there are many works in which data clustering revolves around the use of a calculated correlation coefficient [31, 3]. However, feature detection is not a common use, with such correlations typically utilized in a single step of a longer process [1] or visualized directly [27].

Visualization of many scientific data already represents a reduction of data into screen space. The switch to multivariate visualization dramatically increases complexity which in practice is often shifted to user interaction [36, 8] or multiple linked views [16]. Additionally, in this setting the visualization of multivariate data shares many difficulties with general feature extraction. Considering a transfer function to be a feature extractor, research regarding the design of transfer functions [23, 18] is applicable. More recent work correlates design and use of transfer functions more directly to high dimensional spaces such as statistical [9], attribute [20], or feature [21]. Early [17] and often [7, 24] targeted features in multivariate data were consistent relationships among variables. General feature extraction approaches include discovery such as navigating hierarchical segmentations [13] as well as “creation” as in devel-
Figure 1: For a toy 3D dataset, (a) the percent of data residing in the largest 20% of bins, from 1 to 100 bins. And (b) for histogram sizes from 1 bin to 30,000 bins, the difference between counting empty bins (upper line) and not counting empty bins (lower line).

Metrics are often also employed to evaluate clustering techniques [2, 30]. While we have noted that it is atypical for the clustering-based analysis of climate data to rely upon spatial locations, it is not completely uncommon. Han et al. provide a survey of such spatial techniques [10]. In fact, the histospam shares many similarities with certain approaches such as fuzzy geographically weighted clustering [22]. The defining characteristic of this work is the lack of demarcation between the clustering and “reclustering” based on neighborhood information. That is, we assume no inherent correctness of the initial clustering output, which is treated only as a preprocessing step. In our approach we do not assume nor leverage the existence of a relationship between spatial locality and “neighborhood”.

3 The Histospam

In clustering, spatial dimensions \((X, Y, Z)\) are variables just like any other, and clustering is accomplished over high dimensions by considering every element to be a point in (possibly very) high dimensional variable space. Inclusion of an element into a cell is determined via finding the cluster with a centroid nearest the element in that space. In our approach, we do separate spatial dimensions from the variables of the dataset, as this is the way in which it is natural to measure the difficulty in finding a feature. We therefore consider each spatial location to contain a single value which is also a point in variable space.

In this section, we will first discuss considerations for using histograms alone for clustering. Then we will detail the addition of spatial component to create the histospam. We will end with implementation details of the histospam as an operator in VisIt [4].

3.1 Clustering via Histogram

Using a histogram for clustering is an easier calculation than creating a histogram. For a histogram with \(B\) bins, a data element \(d_i \in D\) is classified by the following bin, \(b\):

\[
b = \left\lfloor \frac{d_i - \min(D)}{\max(D) - \min(D)} B \right\rfloor
\]  (1)
We are now faced with selecting the number of bins, $B$. Finding the optimal number of bins for a histogram is not a trivial task [32], but the accuracy of a histogram is not sensitive to the selection of $B$. For this approach, we are assuming that features are small, we therefore need $B$ to be sufficiently large so that outliers/features have opportunity to be in their own bins. For an initial “guess” we rely on the Pareto principle; eighty percent of all data should exist in twenty percent of all bins. For our toy dataset, the $128 \times 128 \times 128$ single-variable vorticity dataset alluded to in Section 1, we further investigate this choice.

This dataset consists of 2,097,152 data elements, and we varied the number of bins from one to 2,097,152. For each histogram, we calculated the number of elements in each data bin and then what percent of the data resided in the largest 20% of bins. This percentage breaks 65% with a histogram of just 46 bins. See Figure 1(a) to see the percentage over the first 100 bins. With the addition of bins this percentage steadily, but not monotonically, increases over time, maxing out at 75.06% of elements in 20% of bins. However, the rate of increase is extremely slight, with the the percentage remaining under 66% until the histogram has 161,417 bins. This is expected behavior: if 50% of the data is in 20% of bins, a new bin added will decrease this 50% roughly only 20% of the time. Also, since this percentage does not decrease overall, we must add many empty bins. As a sanity check, we perform the calculation while disregarding empty bins which does peak at 65.21% (411 bins), then decreases. Figure 1(b) shows both sets of histograms from 1 to 30,000 bins. The obvious choice for histogram bin size is the aforementioned peak at 411 bins, but the difference between a histogram of that size and the histogram with only 46 bins is negligible. There is therefore no need for the number of bins to differ from a reasonable default for actually viewing a histogram. We set the default to 50 and provide access to the user for tuning this parameter.

### 3.2 From -gram to -spam

A notion fundamental to many clustering techniques (hierarchical, k-means, density, etc) is that distance from a cluster is a litmus test for inclusion in that cluster. We therefore adhere to this while incorporating our elevation of the spatial dimensions. We consider both: an element’s spatial and variable space locations. The histospam introduces a customizable sliding distance calculation to histogram clustering. Specifically, for bins $b_i \in B$ with values and locations, respectively $v_i, l_i$ and similarly for data element $d$ with $v, l$ we want to find $i$ such that the following is minimized:

$$
(1-dw) \times \text{DISTANCE}(v, v_i) + dw \times \text{DISTANCE}(l, l_i)
$$

The distance weight, $dw$, dictates which distance is most important to the calculation. For $dw = 0$ the histospam is a histogram. For $dw = 1$, the clustering is based entirely on how close points are in space (regardless of value). These extremes are shown in Figures 2(a) and (b).

### 3.3 Implementation

We employ a two-pass algorithm. The first pass creates a histogram clustering and calculates each cluster’s (bin’s) centroids in both spaces. The second pass performs the weighted classification. To account for the guarantee of mismatched scales, we normalize the sets of distances themselves before this step.

This algorithm may be parallelized but must include the necessary communications to maintain correct values for cluster centroids as well as the statistics used for the several normalizations. This has been completed; the histospam has been implemented as an operator that works in VisIt’s native data parallel execution pipeline. We have implemented all calculations to work on multidimensional histograms and the user-facing GUI allows for the specification of any number of variables to use in clustering. The user
Figure 2: Visualization of the histospam’s extreme distance weights. A distance weight of 0 is clustering by histogram bin inclusion (a) and a distance weight of 1 leads to spatial regions bunched together (b).

may also specify the number of bins, the distance weight, and the amount of data to throw out thereby leaving only the areas of interest. Upon using the code on climate data we found another user control to add to the interface: the ability to specify a dataset’s fill-value to allow for the easy bypassing of missing values (e.g. those over the ocean in MODIS data). Doing this allows the histospam to provide accurate calculations without any changes to the native grid (the type that can take place with the addition of a threshold operator).

3.4 Weight and Data Selections

Here we describe the histospam as an automatic feature extraction tool. For this to be viable as such, we must be able to auto-tune the distance weight. We do this by performing a sweep over a range of possible weights while maintaining a measure of the change in bin inclusions between iterations. In keeping with our philosophy we select the weight that is most unlike its neighbors under the simple assumption that such a characteristic would make such a weight difficult to find in a trial-and-error process. With regard to how much data to throw away we similarly keep it easy: if a bin has many data elements it is easy to find; we throw away all of the largest bins up to the user-specified threshold. Figures 3(a) and (b) show the result of auto-tuning the histospam on toy data. Figure 3(a) is a rendering of the well known areas of interest for this data. We calculated the percent of data shown in this Figure and input that number as the threshold value for the auto-tuned histospam which resulted in Figure 3(b).

4 On the Simplicity of the Histospam

While much space is devoted to the description of a simple clustering algorithm we believe the strength lies in viewing the problem in this feature-first way. We stated above that any clustering algorithm generates features, but the specific way we defined salient features did not allow for us to couple our design philosophy with more complicated/robust clustering techniques.

As an example, we consider k-means clustering [12] and determine its compatibility in generating salient features as we have defined them. We do not intend for the following discussion to serve as a comparison between the utility of the histospam versus that of k-means, but only to provide insight
into the evolution of the design of the histospam. k-means is a data mining technique for partitioning $n$ observed events into $k$ clusters, with each observation simply belonging to whichever cluster has a mean nearest to the observed value. The result of k-means clustering is a partition of the data into Voronoi cells of roughly equivalent extents, and this is accomplished through an iterative process.

k-means has many applications, such as creating an ecoregionalization of climate data [19]. However, it has some limitations, the main being sensitivity to the choice of $k$. Different choices of $k$ result in different clustering results, and $k$ is often found through a trial-and-error process. There is also another sensitivity, which is how representative points are first chosen to start populating the clusters. Again, different seeds lead to different results. In practice this selection is typically random, though there is research in how to address these sensitivities [29, 33].

### 4.1 Adjustments for “Salient Features”

As stated above, the typical output from k-means clustering is a set of Voronoi cells with similar value extents. The only extents we are concerned with are spatial; we expect features to be small and sometimes near large clusters. It is also possible for a feature to be both few in number and have a large spatial extent, i.e. spread throughout the data. These kinds of features are unlikely to be contained alone in a single cluster contrary to what we would expect out of the “perfect” algorithm. For these reasons, we have no motivation to deploy a refinement process and need only assign cells to clusters once. Additionally, with our focus on creating many clusters of differing sizes, the optimal cluster seeds for our approach are simply a set of equidistant values.

We realized that after some work we had created an algorithm that does nothing more than cluster by calculating a histogram. Fortunately the bins of a histogram actually tend to be well suited to our way of thinking about salient features. However, given that we stripped away all functionality from k-means we had to reintroduce basics in a spatially specific manner. We combine these opposing aspects linearly with a single weight simply to maintain the benefits gained by oversimplifying a clustering algorithm to a histogram.
Figure 4: Auto-tuned histospam on low resolution MODIS data over the continental US. After auto-tuning removes 90% of data (a) remaining regions in Colorado (b) lie within known areas of interest which are colored black (from U.S. Forest Change Assessment Viewer: http://forwarn.forestthreats.org/fcav2/).

Figure 5: Auto-tuned histospam on NDVI from September 2004, 2005, and 2006. There are both strange artifacts due to spatial weighting as well as expected anomalies (from hurricane Katrina damage).

5 Results

5.1 Anomaly Detection

The ultimate goal of much climate research is to find causes of anomalies. However, in many instances, it is necessary to study full series' of ensemble runs in which case there is obvious benefit to simply detecting anomalies. We test the auto-tuned histospam by clustering a large region of data, the continental United States, and then thresholding all but the 10% of data most likely to contain salient regions. This selection adheres to our simple definition of features and is accomplished by simply keeping the clusters with the fewest elements. We have devised two test scenarios based on particular features we would like the histospam to uncover.

As an initial validation of our approach we look at the auto-tuned histogram for a lower resolution 5600m MODIS dataset for NDVI averaged over the year 2008. We are in particular looking to see if a certain region corresponding to damage caused by a widespread mountain pine beetle outbreak in the Southern Rockies [26] remains in the top 10% of data after we deploy the auto-tuned histospam. Indeed, as shown in Figure 4 this known region remains in the auto-tuned histospam. Note, these images are still colored by the original values of NDVI, the histospam results are used for nothing but thresholding here.
### Table 1: Small-scale scalability study of the histospam in terms of bin size, number of variables, and processor count.

<table>
<thead>
<tr>
<th></th>
<th>Bins</th>
<th>Vars</th>
<th>Serial (secs)</th>
<th>2 Cores (secs)</th>
<th>4 Cores (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>-gram -spam</td>
<td>-gram -spam</td>
<td>-gram -spam</td>
</tr>
<tr>
<td>Baseline</td>
<td>25</td>
<td>1</td>
<td>20.16 115.48</td>
<td>10.57 69.13</td>
<td>5.51 39.87</td>
</tr>
<tr>
<td>Changing bins</td>
<td>50</td>
<td>1</td>
<td>20.36 164.78</td>
<td>10.53 100.91</td>
<td>5.59 60.06</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>1</td>
<td>20.49 260.61</td>
<td>10.68 164.22</td>
<td>5.73 100.31</td>
</tr>
<tr>
<td>Changing vars</td>
<td>25</td>
<td>2</td>
<td>28.05 139.38</td>
<td>16.45 83.21</td>
<td>8.78 47.89</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>3</td>
<td>36.55 170.22</td>
<td>21.85 101.32</td>
<td>12.39 60.76</td>
</tr>
</tbody>
</table>

For a second test we used the higher resolution 250m MODIS data across multiple monthly averages. Hurricane Katrina made landfall in New Orleans in September, 2005; as a naive attempt to show associated anomalies we performed an auto-tuned histospam on three variables: NDVI from September, 2004, 2005, and 2006. Figure 5 shows the result of this run. This image is a succinct summary of both the good and bad qualities of our approach. There are obvious distracting selections due to our weighting of spatial locations, but the area damaged by Katrina is in fact represented in the top 10% of this data.

#### 5.2 Performance

As we have mentioned, current automatic tuning involves a parameter sweep which unfortunately can take quite some time. We have also however implemented the histospam to work in VisIt’s data parallel framework and in this section we will give results of some simple parallel tests of the histospam. The following tests are a proof of concept to be used only as a starting point to determine the appropriate use of the histospam. All tests were conducted on an early 2013 MacBook Pro with a 2.7GHz Intel Core i7 processor. The larger 250m resolution MODIS data was used (12Kx12K samples).

We expect the clustering performance to be dependent on both the number of bins used as well as to the number of variables involved. Conceptually, how these would affect calculating a histogram is more easily understood and provided as a frame of reference. These timings are reported in Table 1. The clustering by histogram alone timings are as expected and while the timings for the histospam are significantly slower the overall scaling is encouraging. That is, clustering via histospam is reasonably attainable; more resources help, but by a predictable amount.

#### 6 Conclusion and Future Work

In this paper we presented the histospam, a clustering algorithm that elevates the set of spatial dimensions of a dataset to a level similar to that of its scalar variables. We have shown that this technique is useful in the automatic detection of anomalous regions of the MODIS satellite data. The primary contribution of this work is the presentation of evidence that spatial locations are inherently valuable in the process of feature discovery.

There are many potential directions for related future work. First, a much more comprehensive comparison to other feature detection methods is required. Also, our simple linear combination of spatial and non-spatial components quickly leads to distracting artifacts when increasing the number of scalar variables used in the clustering. That is, there is a need for a better/more complex strategy for distance weighting. Another issue is that unless we assume our auto-tuning is always perfect there is a disconnect...
between weighting selection and interactivity. We would like to explore ways of implementing directed, interactive weight selection.

7 Acknowledgments

This work was completed by staff of the Blue Waters sustained-petascale computing project, supported by NSF Grant OCI 07-25070. We give thanks to Shijie Shu and Forrest Hoffman for help in acquiring and understanding the MODIS datasets. We also gratefully acknowledge the excellent comments and suggestions of the anonymous reviewers.

References

A Feature-first Approach to Clustering . . .


