Multivariate Relationship Specification and Visualization

A Dissertation

Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Roberto R. Sisneros

May 2009

Dedication

For Melissa, my beautiful and loving wife. I adore you.

Acknowledgments

I would like to acknowledge, first, the members of my committee: Drs. **Brad Vander Zanden**, **Jim Plank**, and **Pengcheng Dai**. Thanks for spending some of your valuable time on me.

I would also like to acknowledge the University of Tennessee's Computer Science Department for many many years of financial support as well as providing other necessary resources for the completion of this work. I am grateful to acknowledge this contribution to my education.

To the **members of SeeLab**: balancing both the thought-provoking and ridiculous with you guys over the years has made me the computer scientist I am today.

Dr. Jian Huang: Thanks for introducing me to Visualization, and then sharing some of your abundant brilliance and wisdom with me. I would have never made it this far without someone like you to fall back on.

Mom: Thanks for keeping me aware of my family's constant love and support, even from a distance.

Grandpa: Thanks for teaching me Archimedes' Principle and for diagramming why perpetual motion machines just won't work while I was still practically in diapers – it was just the push I needed.

Abstract

In this dissertation, we present a novel method for multivariate visualization that focuses on multivariate relationships within scientific datasets. Specifically, we explore the considerations of such a problem, i.e. we develop an appropriate visualization approach, provide a framework for the specification of multivariate relationships and analyze the space of such relationships for the purpose of guiding the user toward desired visualizations. The visualization approach is derived from a point classification algorithm that summarizes many variables of a dataset into a single image via the creation of attribute subspaces. Then, we extend the notion of attribute subspaces to encompass multivariate relationships. In addition, we provide an unconstrained framework for the user to define such relationships. Although we intend this approach to be generally applicable, the specification of complicated relationships is a daunting task due to the increasing difficulty for a user to understand and apply these relationships. For this reason, we explore this relationship space with a common information visualization technique well suited for this purpose, parallel coordinates. In manipulating this space, a user is able to discover and select both complex and logically informative relationship specifications.

Contents

List of Figures

Chapter 1

Introduction

It is common for scientific visualization production tools to provide side-by-side images showing various results of significance. This is particularly true for applications involving time-varying datasets with a large number of variables. However, application scientists would often prefer to have these results summarized into the fewest possible images. In this work, we are interested in developing a general scientific visualization method that addresses this issue. In particular, we summarize multivariate data for previewing, i.e. we provide general information about a dataset that a user has a limited knowledge of.

Our method produces single images that fuse key aspects of multiple attributes of the data. From those images, one can discover prominent inter-variable relationships, even though the renderings are independent among variables. Our approach also produces simple statistical evaluations of the data that can be used to effectively navigate the dataset. For instance, which timesteps are important to examine, and what particular variables in those timesteps deserve special attention.

We demonstrate this approach by visualizing SciDAC [DoE, 2000] caliber datasets from global coupled climate-carbon cycle model simulations produced by different models. We draw our research motivation from the current thrust to generate accurate simulations of the global carbon cycle that model the interactions and feedbacks between the terrestrial biosphere and the climate system.

The visualization aspects of this task are very demanding for several reasons. Firstly, there are a large number of variables involved in each simulation. Exacerbated by the need to study

multiple runs in a cohesive manner, the combinatorial space that needs to be explored is overwhelming, even just the task of studying two variables from two simulation runs. For instance, scientists already have some empiric understanding of how net ecosystem exchange relates to net primary productivity. Does the relationship exist as expected in a peta-scale simulation, including across different time spans of different runs? This model of investigative study and the need of high interactive rates currently present a challenge for large data visualization.

Knowledge extraction from multivariate data is a pressing problem of visualization research. A common user need is to understand how multiple variables relate to each other (i.e. multivariate relationships). So far, there have been few existing methods that can effectively visualize multiple multivariate relationships in the same setting. To address these issues, we extend the notion of attribute subspaces to incorporate multivariate relationships. Our approach uses a universal discrete representation of multivariate relationships and lifts the constraints on types of relationships considered. Human color perception is the practical limit on the number of multivariate relationships that can be simultaneously considered. By classifying every point in the volume as belonging to one of the relationships, we can show a single view of the data that summarizes the existence of all specified relationships. To aid the user in the discovery of previously unknown variable dependencies, we spatially catalogue the most intuitive relationships between two variables. These are provided as default settings in our system.

We find that with a system in place for the visualization of multiple multivariate relationships, we quickly encounter complex and inexplicable results when attempting to analyze relationships among several variables. For this reason, we have developed a method for analyzing these relationships by utilizing parallel coordinates for high dimensional analysis. In this work, we present an approach for the classification of parallel coordinate plots via image space metrics. We propose several such metrics, each identifying parallel coordinate visual properties. We then detail a system for interactively sorting and navigating large sets of parallel coordinate axis pairs by leveraging these metrics. Lastly, we implement a directed decluttering scheme that aims to extract visual features from cluttered parallel coordinate renderings based upon metric values of a parallel coordinate plot with a desirable set of visual properties. We demonstrate this approach using a public domain dataset of time-varying climate simulation.

In the remainder of this work, we first describe the related work of this research in Chapter 2. In Chapter 3, we present our design and implementation of the overall system, followed by an ap-

plication study in Chapter 4. We discuss the framework for relationship specification in Chapter 5. We show the principles of our analysis scheme in Chapter 6, and conclude in Chapter 7.

Chapter 2

Literature Review

2.1 Multivariate Visualization

When dealing with multivariate data, the most straightforward approach is to treat each variable separately. However, that approach is too simplistic and still leaves much to be desired. The visualization research community has undertaken the task of finding more capable methods. In particular, current literature usually falls within two categories: fusing multiple variables into single images [Bair et al., 2006, Kirby et al., 1999, Riley et al., 2003, Helgeland and Andreassen, 2004], or viewing relationships between variables [Kniss et al., 2001, Sauber et al., 2006]. A great survey of many of such techniques has been presented by Wong and Bergeron in [Wong and Bergeron, 1997a].

Creating images from many variables is a hard problem, as shown in [Taylor, 2002]. It is often impossible without somehow reducing the data of each variable. One way to do this is to focus on only the salient regions of each, namely by feature detection and extraction. If features can be identified and highlighted, it becomes much easier to render features of different variables concurrently. [Love et al., 2005, Walter and Healey, 2001, Weiler et al., 2005] have presented a number of successful example methods based on feature extraction. A common limitation of these methods, however, is the a priori requirement to have features accurately defined before a visualization can be created. To address this limitation, Jänicke et al. recently developed a region detection method that is both application-independent and that doesn't rely on user input [Jänicke et al., 2007]. Their approach extends local statistical complexity of cellular automata to perform region detection in

applications that can be described by PDEs.

Woodring and Shen proposed chronovolumes as a technique for displaying time varying volumetric data [Woodring and Shen, 2003]. By treating different timesteps as different variables on the same voxel, their approach also provides some inspiration to multivariate data visualization. Images created by chronovolumes contain a combination of all timesteps of interest, while maintaining a context of surrounding earlier and later timesteps. This is done by adding an integration through time to the raycasting pipeline. They also propose a similar coloring scheme in which they simply assign each timestep its own color. They suggest 20 as a limit on the number of discernible colors. This is an existing example that demonstrates the power of attribute-specific subspaces. However, in this case, it is specific to the dimension of time.

Multivariate visualization is a unique area due to the large number of variables. This is compounded when timesteps are considered as independent variables and the combinatorial space to explore is exponentially large. This special research need of multivariate visualization calls for methods to provide summarizing previews of data. From this respect, our method is based on logical operators that can be combined in a customizable manner. These basic operators provide a way to visually gauge relative strength of variables, and hence guide a user's attention to a much reduced subset of an otherwise large and incomprehensible multivariate dataset. In [dos Santos and Brodlie, 2004], dos Santos and Brodlie introduce a conceptually similar approach to handle multivariate and multidimensional data. In their work, a filtering, or subsetting, operation is used to select a subset of all variables and dimensions to create a smaller space from the original high dimensional space. Woodring and Shen further extended the idea to include set operations [Woodring and Shen, 2006]. A set can be specific to any variable as a boolean relationship. Sets can be combined using set operations in an effort to select voxels of interest to the user.

In both of these representative works, selection choices are decided and applied uniformly for the entire dataset. Our work is different in that the variable or attribute we show is eventually a per point selection. In addition, we do not consider spatial dimension for subsetting. In other words, the subspaces we create are dependent on both attributes and points.

Finally we must note that many pioneering researchers have studied how to provide succinct and informative visualization of multivariate data through non-photorealistic and illustrative approaches. There, the focus of visualization research has been to develop innovative visual cues inspired by artistic as well as real world domains [Healey and Enns, 2002]. Common technical

approaches in the domain of non-photorealistic rendering include texture synthesis [Lu et al., 2007, Weiler et al., 2005], glyphs [Healey and Enns, 2002] and stippling effects [Lu et al., 2003], etc. However, in those works, the foci are features that are well understood (including the definition of the features and the most effective visual analogies for those features). The starting point of our work is different in that our focus is to develop basic operators that can be used to define attribute-specific subspaces and to demonstrate their usefulness. Current sophisticated and powerful illustrative rendering techniques can directly leverage our definition of attribute-specific subspaces for better results.

2.1.1 Multivariate Information Visualization

A focus of this work is to uncover the existence of multivariate relationships in time-varying volume simulation data. To the best of our knowledge, this topic is still relatively novel. However, we notice a number of pioneering researchers have addressed similar problems in the realm of information visualization.

As surveyed by Wong and Bergeron in their 1997 work [Wong and Bergeron, 1997a], many successful techniques had already been developed for multivariate information visualization. Some of the more prominent approaches include parallel coordinates [Inselberg and Dimsdale, 1990], creative design and use of shape, color and textures [Beddow, 1990, Levkowitz, 1991], dimension stacking [LeBlanc et al., 1990] and hierarchical axes [Mihalisin et al., 1991a, Mihalisin et al., 1991b]. More recent works in this field very often focus on more complex tasks such as visual cluttering caused by dense data. We categorize these recent approaches as one of the following: (i) creating innovative layouts of data abstractions (e.g. association rules [Wong et al., 1999]) in visual display and the corresponding new methods of user interaction, (ii) computing derived features of the data and producing summarizing views based on the derived features [Wong and Bergeron, 1997b] and (iii) data fusion, where often multiple see-through layers of visualization are overlaid to create a final visualization [Wong et al., 2002]. In [Draper et al., 2008], a visual query language is presented for identifying relationships.

Our work is similar to many of these in that our goal is the identification of relationships using many of the same tools. However, we strive to maintain canonical volume visualization principles in our approach, i.e. the spatial integrity of the underlying data must remain prevalent. For this

reason, we cannot directly use these methods. In this context and with these restrictions, the volumetric and dense nature in our data severely limits our choices of layouts and visual metaphors. However, we still drew valuable inspirations from the information visualization methods, particularly in regard to the potential of creating summarizing views based on (i) visual fusion of the information [Wong et al., 2002] and (ii) the additional analytical results of the data [Wong and Bergeron, 1997b].

2.1.2 Multivariate Volume Visualization

In the original framework of volume visualization by Drebin et al. [Drebin et al., 1988], volume visualization is inherently multivariate. The pivotal component is the transfer function. However, how to design a good transfer function in general settings is still an open problem. Scout [McCormick et al., 2004] is likely the most versatile system for runtime design and customization of transfer functions on the fly, and as in our approach, Scout [McCormick et al., 2004] provides a programming language interface. Their language is well suited for deciding the visual attributes of various parts of the data to achieve the most distinctive rendering results. Our language works well for specifying relationships and classifying the volume accordingly. We believe the two languages to be very complementary.

While it is hard, finding the consistent existence of relationships carries a huge impact in our field. Typically, the approach taken is one of moving from domain knowledge to the discovery of a plausible relationship: examples include gradient based [Kindlmann and Durkin, 1998], curvature based [Kindlmann et al., 2003], and importance-based [Viola et al., 2004] methods. Our work is different in that we aim to work in the opposite direction, from raw data to the discovery of novel domain knowledge.

There is also a recent trend to achieve interactive runtime data reduction by doing compound queries [Glatter et al., 2006, Stockinger et al., 2005]. In a way, this is actually about relationships, just one at a time. There has been a difficulty in scaling beyond the limited number of relationships that are easily concurrently viewable. From this respect, our method provides a way to visually examine more relationships together, while directly using query-based techniques to generate complex and meaningful relationship specification files.

As a few representative examples, existing methods to view relationships include using pro-

cedurally specified high dimensional Gaussian functions [Kniss et al., 2002], or computing and rendering correlation coefficients between variables [Gosink et al., 2007]. In those systems, the primary focus is the strength of the single type of the relationship of concern. Our work differs in that we allow more generic classes of relationships to be evaluated, and we consider the type of relationship present as the primary issue. On a voxel by voxel basis, different types of relationships compete for representation in the final rendering. Strengths of relationships are still considered but are of a secondary role.

Adding a competition based concept into transfer function design has been tried in an implicit way before, for instance when using neural networks for volume classification [Tzeng et al., 2003]. Our method is explicit and can help to make the classification process a clear-box, instead of a black-box.

There is also work to achieve interesting multivariate results by problem space simplification. One way to do this is to render multiple values on a surface, however, as shown in [Taylor, 2002], this approach still does not scale beyond a handful of variables. Woodring and Shen presented work on using a different color to represent different neighboring time steps of the same simulation [Woodring et al., 2003]. They focused on showing various statistical features along the temporal axis. This work compliments our belief that even limited to just using colors, there is still a lot of information that one can show.

Lastly, multivalue data visualization has also recently been reported in the literature [Love et al., 2005]. Multivalue data are those that have multiple instances of the same variable recorded on each data point. While our method could potentially be extended to handle multivalue data, multivalue visualization is beyond the scope of this work.

2.1.3 Parallel Coordinates

Parallel coordinates [Inselberg, 1985] represent an analogue technique to Cartesian scatterplots [Moustafa and Wegman, 2002] and are used to intuitively analyze high-dimensional data in a 2D plane. A parallel coordinate plot highlights data parallelism by placing the plot axes parallel to each other rather than placing them orthogonally. A clear benefit to parallel coordinate plots is the ability to string together an arbitrary number of axes and maintain a similar intuitive context to few axes, whereas with orthogonally placed axes, i.e. scatterplots, analysis in more than three dimensions

becomes exponentially difficult.

In addition to the above properties, it has been shown that many properties of scatterplots hold true for parallel coordinates as well [Inselberg, 1985, Inselberg and Dimsdale, 1994]. Typical areas of research in the field of parallel coordinates address providing meaningful axis orderings and reducing visual clutter in axis pairs. Also, although early work in axis ordering [Wegman, 1990] proposes techniques for axis selection, the problem of finding an optimal layout, even a data dependent one, remains an open problem in this field. This is an especially difficult issue when dealing with large data.

In [Ellis and Dix, 2007] the authors present a taxonomy of visual clutter reduction techniques. These are first classified by general techniques used. They define sampling [Derthick et al., 2003, Ellis et al., 2005, Rafiei and Curial, 2005] as "the random selection of a subset of data" and filtering [Ahlberg et al., 1992, Ahlberg, 1996, Brodbeck et al., 1997] as "selection of a subset of data that satisfies a given criteria." Our approach is in essence a filtering approach, however there is no clear criteria for any one line in a rendering to determine whether or not it is to be filtered, this is dependent on the order in which lines are selected, and we select them randomly. For this reason, we feel our approach is best considered a hybrid of these two techniques. Secondly, all techniques are classified in terms of attempted goals each strives to achieve. Our approach readily falls under the category of "avoids overlap" and within that, the subcategory of "ability to see/identify patterns" as in [Ahlberg and Shneiderman, 1994]. Other hybrid works exist that provide interactive systems for a user to try out one or more techniques on a trial-and-error basis [Ellis and Dix, 2006, Steed et al., 2007].

In [Enrico Bertini, 2006] the authors present a sampling method that relies on metrics to define the quality of images to drive corrective actions. They address the issue of whether or not sampling is too weak or too strong in terms of remaining densities. However, in our case, if there is interesting data in the original dataset, very high or low density may be desired. Their decluttering scheme, although based on density, is in essence a directed decluttering approach, in that they're stopping criteria is based on reaching a target density. Our decluttering is based on using metrics classifiers rather than measurements of "goodness" (although they may still be used in this manner), and is therefore directed toward specific visual properties, with its overall performance directly dependent on how well the metrics describe those visual properties. One work that uses metrics in a different context [Johansson et al., 2008], uses metrics as a measure of visual

quality to analyze how well users deal with noise in renderings.

There are also examples reducing clutter by axis redirection, axis reordering, clustering or subsampling [Wegman and Luo, ,Peng et al., 2004,Fua et al., 1999,Johansson et al., 2005b]. In [Ellis and Dix, 2006] the authors provide a justification for random sampling as a decluttering tool, and based on the tradeoff between accuracy and performance conclude it's the optimal strategy for decluttering within a sampling lens. This is a notion we leverage in our decluttering scheme.

There have been several augmentations to traditional parallel coordinate plot rendering. In [Zhou et al., 2008] curved lines are favored over straight-line techniques. In [Graham and Kennedy, 2003] the authors attempt to disambiguate areas with many crossings or shared values using Gestalt principles. Another work, [Novotny, 2006], uses an outlier preserving method to render parallel coordinate plots that utilizes binning to determine relative strengths of trends in the rendering. There has also been work in circular [Ferreira and Levkowitz, 2003] and 3D [Johansson et al., 2005a] variants of parallel coordinate plots. The authors of [McDonnell and Mueller, 2008] propose illustrative parallel coordinates, a set of multiple rendering techniques such as edge bundling and opacity/shading effects to improve parallel coordinate renderings.

This work differs from existing work by taking the study of metrics in parallel coordinate plots in a new direction. Specifically, we take a fresh look at what can be accomplished by metrics within this framework, and with this new understanding attempt to improve existing methods that leverage metric calculations. We also strive to make this process flexible in terms of user preference by providing an interface for customizability and offering results that provide feedback for areas of improvement.

Chapter 3

Attribute Specific Subspaces

In this chapter we present a point classification algorithm for multi-variate data. Our method is based on the concept of attribute subspaces, which are derived from a set of user specified attribute target values. Our classification approach enables users to visually distinguish regions of saliency through concurrent viewing of these subspaces in single images. We also allow a user to threshold the data according to a specified distance from attribute target values. Based on the degree of thresholding, the remaining data points are assigned radii of influence that are used for the final coloring. This limits the view to only those points that are most relevant, while maintaining a similar visual context.

3.1 The Approach

Our prototype system is named massSIV (multiple attribute specific subspace visualization). We intend for massSIV to be generally applicable, however, it is currently designed primarily for scientific simulations that produce volume data. We take as input a general dataset that has, at each location, an associated attribute set. Each attribute must have a value at each location (point). Also, the user supplies a set of attribute target values that allow us to determine for any value at any location, whether or not that value is considered a "good" one, i.e. close to its target value. Our goal is to then provide a rendering of the dataset that is colored based on all attributes. We first split the colorspace, to associate with each attribute its own color. Then, for every location, we select a single value, the one closest to its attribute target value. However, if no values are close to

their target values, we may threshold that point. We then redistribute the open spaces created by thresholding to the remaining points, giving each an area of the final space that it may color. After a coloring is chosen, any set of points with the same color is an attribute subspace. The selection of a volume's attribute set is arbitrary, therefore there is a wide variation, and hence flexibility, in the combination of attribute subspaces.

3.1.1 Attribute Set Selection

A typical time-varying dataset contains a set of variables associated with each point. The obvious choice for an attribute set is to simply use the variables already associated with each point, and render a single timestep. Reduction of the attributes allows us to see salient regions for all variables at once. We can also focus the attribute set on the timesteps of a dataset. Here, the attribute set for each point would be a single variable's value across many timesteps. In this case, we show, for a variable, in which timesteps interesting features arise. If interest lies with a certain variable, this could be used to help determine which timesteps would be most valuable in that variable's evaluation. For this reason, we rank all attributes based on the percentage of the image they color. Similarly, we could reduce the spatial dimensionality of a dataset. The attribute set would be, for a 2-D slice of the volume, a certain variable's value across the remaining dimension. The resulting rankings could then be used to determine a slice of the volume to focus on. In the following sections we assume the attribute set is a point's variables.

3.1.2 Creating Attribute Subspaces

Now that the dataset has an attribute set, i.e. each point has associated with it a set of values, we create attribute subspaces. Each value at every point corresponds to one of the attributes in the attribute set. As stated, we now select a single value that is closest to its attribute target value. However, this is not relative to values of a point, but to all the values of an attribute. We do this by calculating the set of distances from each value to that attribute's target value. Then, for every value of a point, we calculate how far away from the median of the distance set that value is. We then select a point's single value to be the one from the initial set of values that is both close to the target value, and farthest from the median. By doing this, we ensure that a value is "good" relative to the entire set values of an attribute, i.e. we say that value has a high impact on an attribute at

that point.

Along with a point's single value, we also store which attribute that value came from, in the form of an integer attribute ID. The final dataset we create will contain only such attribute IDs, however, based on neighboring points' values, a point's current attribute ID may change. We call the set of points colored by a specific attribute an attribute subspace of the volume. We realize that often it is impossible to define a set of target values since it is unknown which values are interesting. If no target values are specified we take the highest values for each attribute to be the target values. We also allow for average values to be used for the target values, in the case that the high values of attributes are outliers. Figure 3.1 details a simple example of creating attribute subspaces.

3.1.3 Thresholding

We consider salient regions to be those in which attribute values are near target values. For this reason, thresholding in massSIV is not a simple filtering, but is done by removing points that contain no values that are near their attribute target values. A high level of thresholding then focuses on only those values closest to the attribute target values, and therefore the most salient regions. The user may specify a distance from the target values to use as a threshold. By default, we threshold all points that contain no values that are closer than the median of the distance set from the attribute target values. In Figure 3.1 P3 is thresholded.

3.1.4 Redistribution of Open Spaces

Although thresholding highlights the points closest to attribute target values, we would also like to maintain a regional context in the case of few remaining points. To achieve this, we redistribute the open space created by thresholding to the remaining points. We give each remaining point its own radius of influence. Firstly, for each attribute we calculate an attribute volume percentage. This is the number of remaining points associated with that attribute divided by the total number of remaining points. We then normalize the values for each attribute, making each point, P, the value that corresponds to a percentage of its attribute, $norm(P)$. A point's volume, volume(P), is then that point's percentage of its attribute's volume percentage. To find a point's radius, we

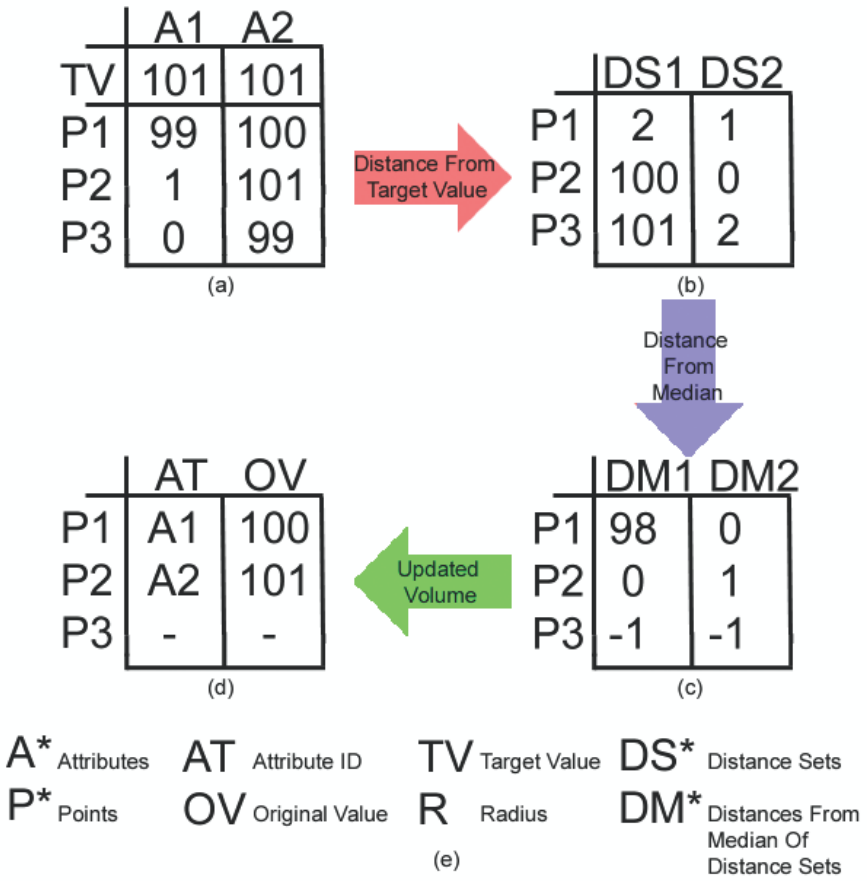


Figure 3.1: (a) Simple 3 point dataset. (b) The distance from target values (distance set) for each attribute. (c) The distance from median of distance set: notice P1's second value is closer to its target value, but its first value is chosen because it is much closer relative to how close the rest of the points' first values are to A1's target value. (d) The new volume, the starting point for the redistribution of open spaces. (e) Key for Figures 3.1 and 3.2.

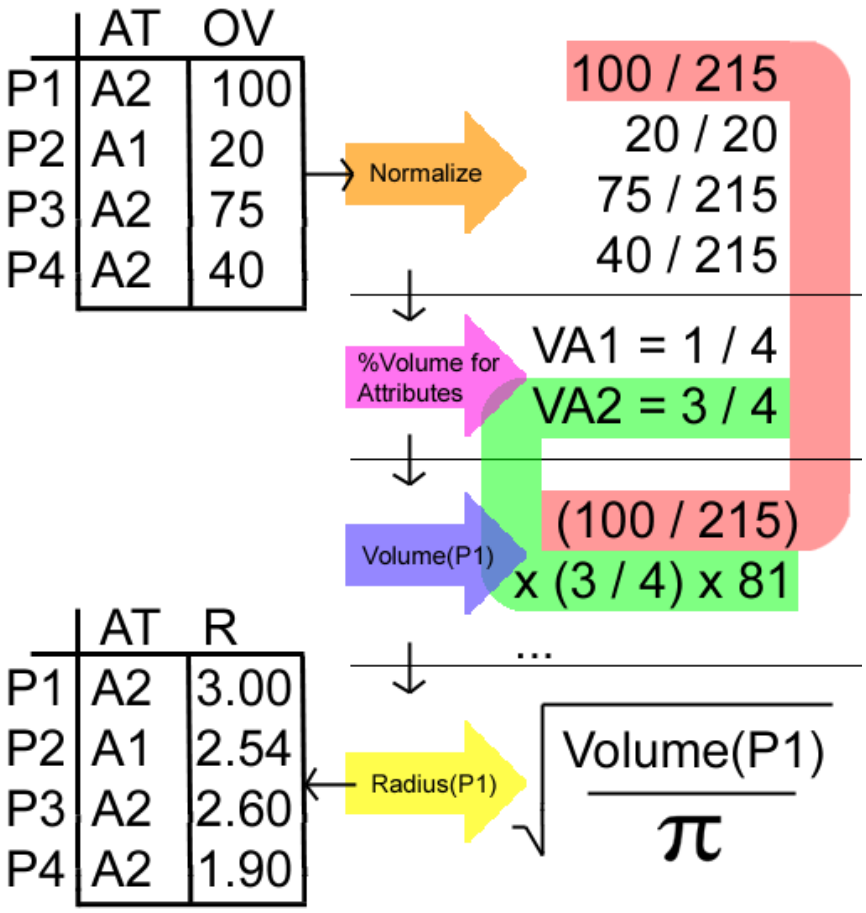


Figure 3.2: Detailed calculation of P1's radius of influence. In this example, there are only 4 points remaining after thresholding a a 2-D 9x9 dataset (81 is the total area of the space). Note: Equation 3.1 is for radius calculation in 3-D.

simply set a point's volume equal to the volume of a sphere and solve for the radius:

$$r = \sqrt[3]{\frac{3 * volume(P)}{4\pi}} \tag{3.1}$$

We denote the radius of a P as $radius(P)$. An example of radii calculations is provided in Figure 3.2.

3.1.5 Finalizing the Attribute Subspaces

To maintain a sense of spatial locality, we have a point's influence over another point dissipate within it's area of influence, the farther away the two points are. We achieve this with a simple linear function. For two points, P_1 and P_2, $dist(P_1, P_2)$ is the euclidean distance between those points. For a point P_2 that is within P_1's area of influence, P_1's influence on P_2 is then:

$$f(P_1, P_2) = norm(P_1)\left(1 - \frac{dist(P_1, P_2)}{radius(P_1)}\right) \tag{3.2}$$

This value is calculated between each point, P, and all other points that contain P within their radii. P then receives the value of the point containing P with the maximum value of Equation 3.2. The final output of our program is a spatial dataset where each location is reduced to a single integer tag that corresponds to one attribute, and therefore one color. In the case that a point falls within no other points' radii of influence, that point is assigned the value zero, which in our rendering is represented by black (2-D): the background color, or clear color (3-D): fully transparent. This dataset is the completed set of attribute subspaces.

3.2 Rendering Attribute Subspaces

2-D images created from our resulting datasets are simple bitmaps created from a 2-dimensional slice of the volume, with each point colored by its attribute color. To visualize the volumes resulting from our method, we have implemented an hardware-accelerated volume renderer for interacting with tag volumes. The unordered and discrete nature of tags makes it impossible to use traditional linear interpolation for determining a sample's tag, as it may introduce false intermediate tags between segments. Nearest neighbor interpolation, on the other hand, produces

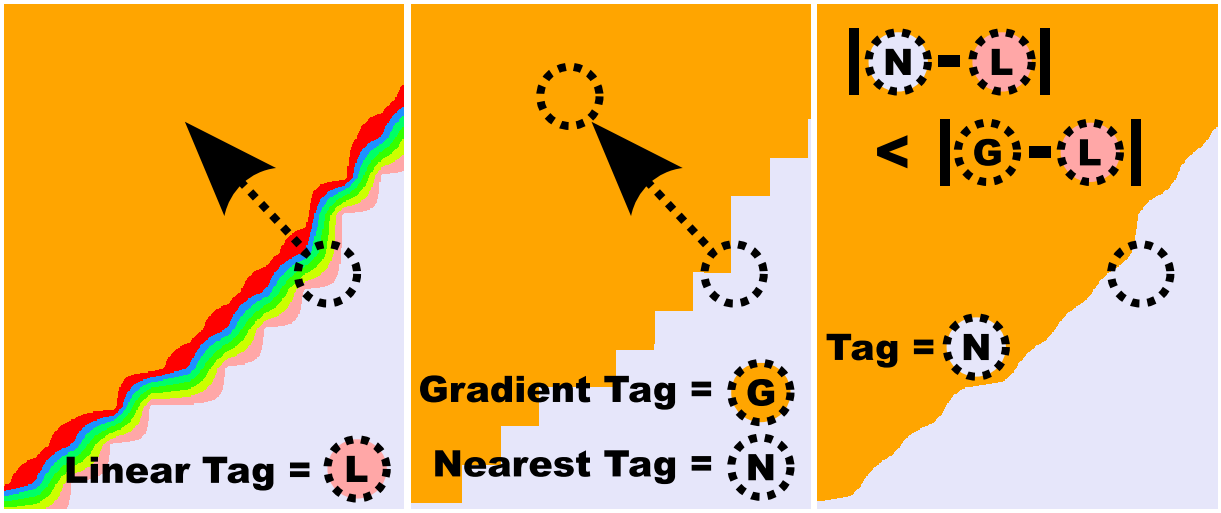
correct results but the voxel resolution is immediately apparent and can detract from the visualization.

Most previous approaches to improved boundary filtering of tagged volumes rely on several passes [Tiede et al., 1998] or a maximum of four tags [F.Vega et al., 2005] for correct interpolation. To avoid these limits, we have developed an interpolation scheme based on a sample's proximity to a gradient-aligned neighbor. We evaluate this proximity using the sample's linearly interpolated tag to choose between the sample's nearest tag and the gradient neighbor's nearest tag. This method requires two textures to be sent to the GPU. The first texture contains only the tags and uses a nearest neighbor filter. The second texture contains both the tag gradients and the tags, using a linear filter. For the gradient calculation, we found central differencing to produce gradients of poor quality across tag boundaries [F.Vega et al., 2005]. Instead, we compute the tag gradients using linear regression over a 3x3x3 neighborhood [Neumann et al., 2000].

For the rendering, view-aligned slices are rendered in back-to-front order, and a fragment program is invoked for each fragment across a slice. To determine a sample's tag t at subvoxel resolution, we use the gradient to locate the gradient-aligned neighbor, i.e., the nearest neighboring voxel to the sample along the gradient direction. In most instances, the value of the tag around the sample will range from the nearest tag t_n to the gradient tag t_g, with a smooth boundary lying halfway between t_n and t_g. We use the linear tag t_l to locate this halfway point. To compute t smoothly, then, we examine the difference between t_l and the other two tags and set t to the closer. Figure 3.3 illustrates an example where t_n is closer to t_l. As shown in the following equation, if t_l is closer to the nearest tag, $t = t_n$. Otherwise $t = t_g$.

$$t = \begin{cases} t_n & : \quad |t_n - t_l| < |t_g - t_l| \\ t_g & : \quad otherwise \end{cases}$$

The computed tag then indexes into a colormap to retrieve the sample's color and opacity. A simple interface allows users to alter each tag's color and opacity. With a volume size of 120x480x512, a 512x512 viewport, and a sampling rate of 1.0, we achieve a frame rate of 8-11 fps on a Dell Precision 470 workstation using an NVIDIA Quadro FX 3450 GPU.



(a) Linear Filter (b) Nearest Filter (c) Final Tag

Figure 3.3: To avoid the inaccuracy of linear filtering and the coarse appearance of nearest filtering, tags are interpolated at subvoxel precision with a hybrid filtering approach. (a) The tag and gradient are retrieved using linear interpolation for the encircled fragment. (b) The tag is retrieved for the same fragment using nearest filtering. Additionally, the nearest tag along the gradient is retrieved. (c) The final filtered tag is determined by using the linear tag to find the closer of the gradient's and fragment's nearest tags. In this case, the nearest tag is chosen.

3.3 Demonstration of the Approach

We reduce multiple images, each representing a data attribute, into a single image, while maintaining discernible features. However, the root of our approach is not in highlighting the most salient regions of the dataset, but to do so at every point. Just as single variable renderings are independent of each other, so are attribute subspaces. However, through thresholding, we do focus on the areas of the volume where attributes have a high correlation to the attribute target values. In the extreme thresholding case, we show only the single points that correspond to the values closest to the attribute target values. Thresholding does highlight regions of interest in a more global sense. Using a point's radius of influence, we also accentuate the area of the volume likley containing events surrounding those highest impact points. We show results from three datasets, the first is a 480x720x120 jet combustion dataset containing five variables, we use only timestep 116. The second, a 256x128 climate dataset containing 7 variables, again we only use one timestep (corresponds to the year 2000). Lastly, a 128x64 climate dataset containing one variable: a CO_2 measurement taken hourly over several years, we use the first hour of Jan. 1 for ten years, 1890-1899.

3.3.1 Concurrent Views

The typical use of our program is to provide a single image showing salient regions of multiple attributes as apposed to showing multiple side-by-side images.

In Figure 3.4 we show the single variable renderings, of a single slice of the volume, of the jet combustion dataset along with the single image resulting from our approach corresponding to the same slice. We did not change the natural coloring of ncBrowse, and in those images, the points with high values are colored in shades of red that clearly stand out. For this reason, we ran our program with the variables' highest values used as the attribute target values. The regions of high values in each single variable image correspond to one of the five colors in the image resulting from our approach.

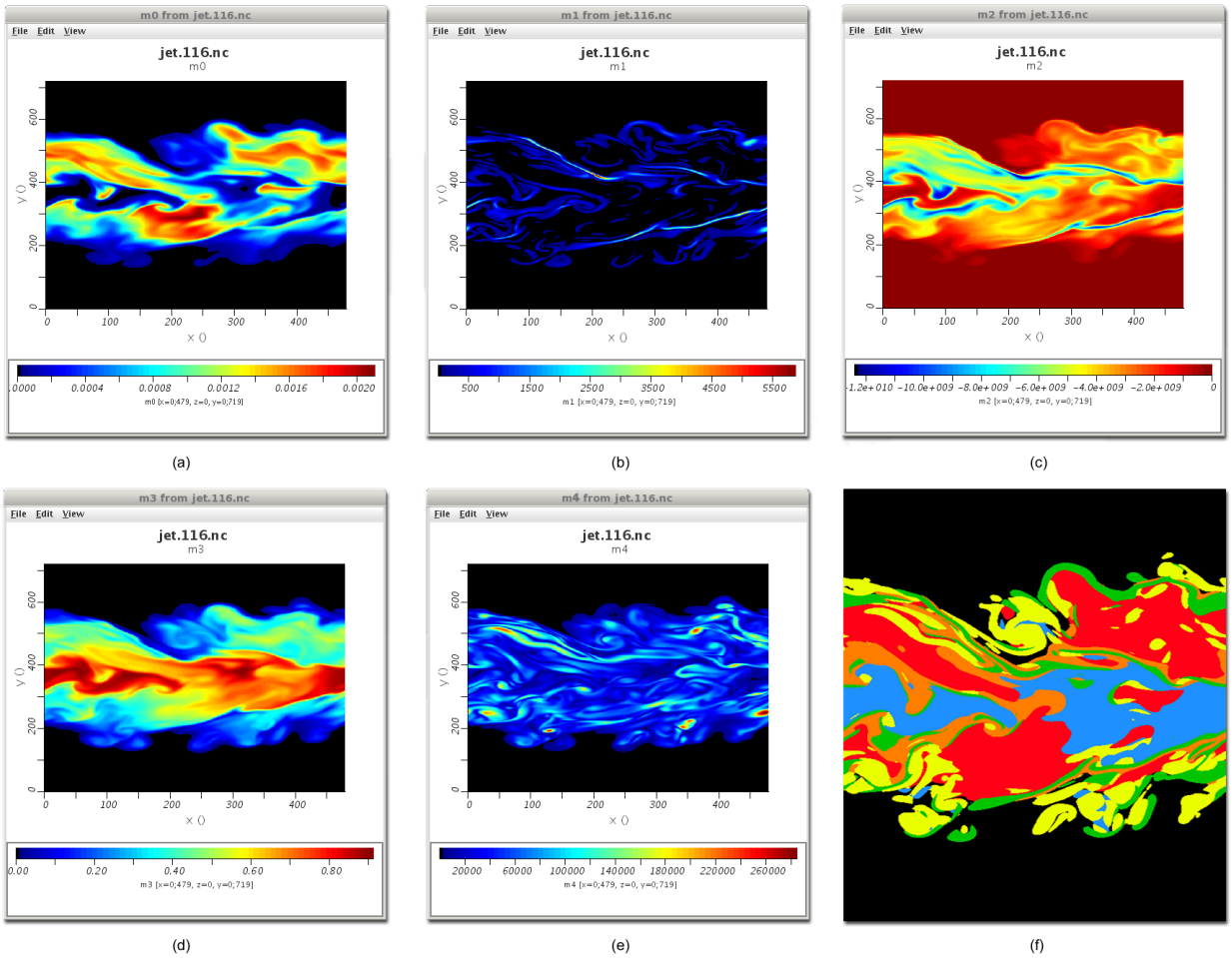


Figure 3.4: Images we created using ncBrowse (http://www.epic.noaa.gov/java/ncBrowse/) from single variables of the jet combustion dataset (a)-(e). Image created by our method fusing high valued ranges of each of the single variable images (f).

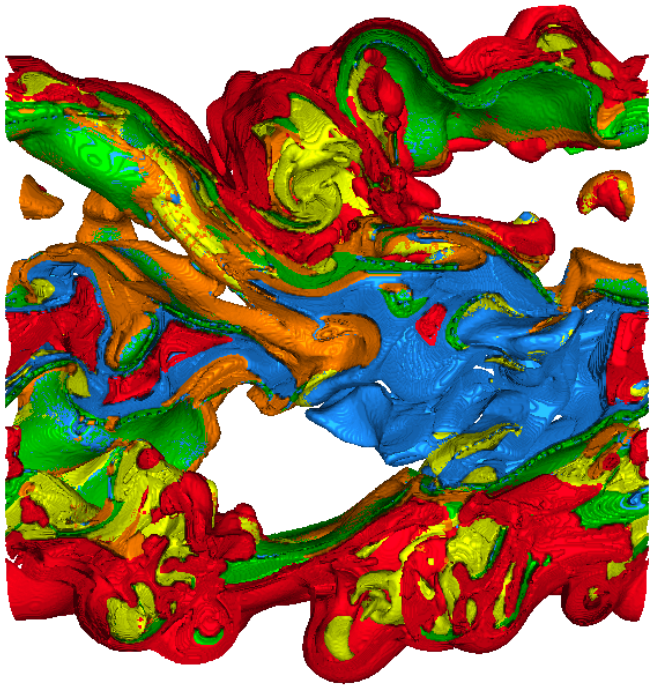
Also from the jet combustion dataset, we show 3-D renderings of the dataset that is the output from our program. In Figure 3.5, we show the results of our program from using both maximum values as attribute target values and average variable values as the attribute target values. Figure 3.5(b) is simply the 3-D version of Figure 3.4(f), and therefore a summary of 5 3-D single variable renderings.

3.3.2 Highlighting Salient Regions

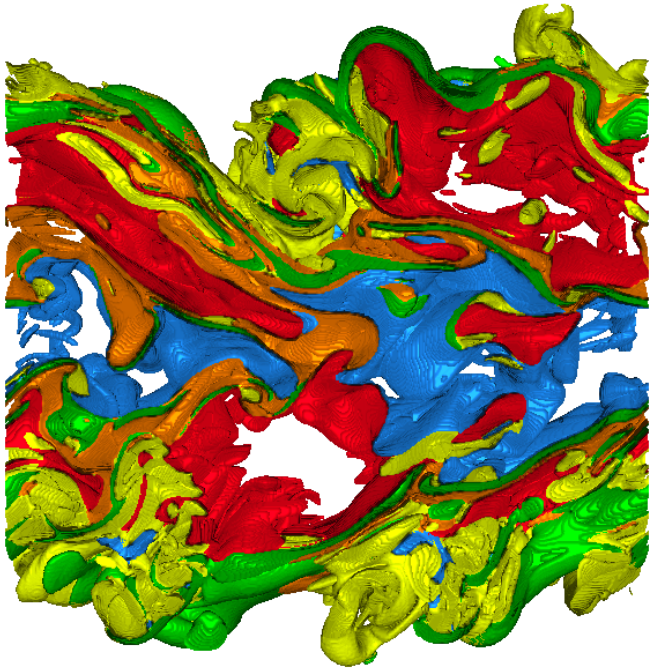
We present two examples of using thresholding to highlight salient regions. In Figure 3.6, the output from our program ran on the jet combustion dataset is colored by only 4.7% of the number of points in the initial dataset. The radii of influence highlight regions associated with the points not thresholded, which is especially important for those small regions. This dataset was also created using the attributes' maximum values as the attribute target values.

In Figure 3.7 we show our program's default output on the 7-variable climate dataset (2-D). To clearly illustrate the thresholding/area of influence relationship, we present the second image of Figure 3.7. This image has the maximum level of thresholding, all but 5 points are thresholded. In this image we highlight the areas surrounding those points, this is an indication of the general areas that are likely to be of interest for those attributes.

Figure 3.7 is a representative example of the utility of massSIV. Specifically, this visualization shows the most extreme variable among the 7 variables on each geographical location. The target audience for such a viewing are those who do not already have an in-depth understanding of the problem domain; our image illustrates the following information. The black regions represent places where all 7 variables are not sufficiently close to the target values, and have been thresholded. Examples of such regions include the flat lands in Africa, north-east of China, central regions in U.S. and parts of Russia. The specific humidity is high over the tropical oceans because the humidity is high in these warm regions over water. Temperatures are high in the South-Pacific (South-Pacific warm pool) and off the coast of Europe due to the Meridional Overturning Circulation. PBL (planetary boundary layer) height is low only over the oceans where other factors are not dominant. Stddev of orography is high in mountainous regions, often forming a ”bulls eye” around surface geopotential, where the altitude is high enough to dominate the stddev of orography. Surface pressure is low in the Arctic and high in the southern oceans due to wind circulation

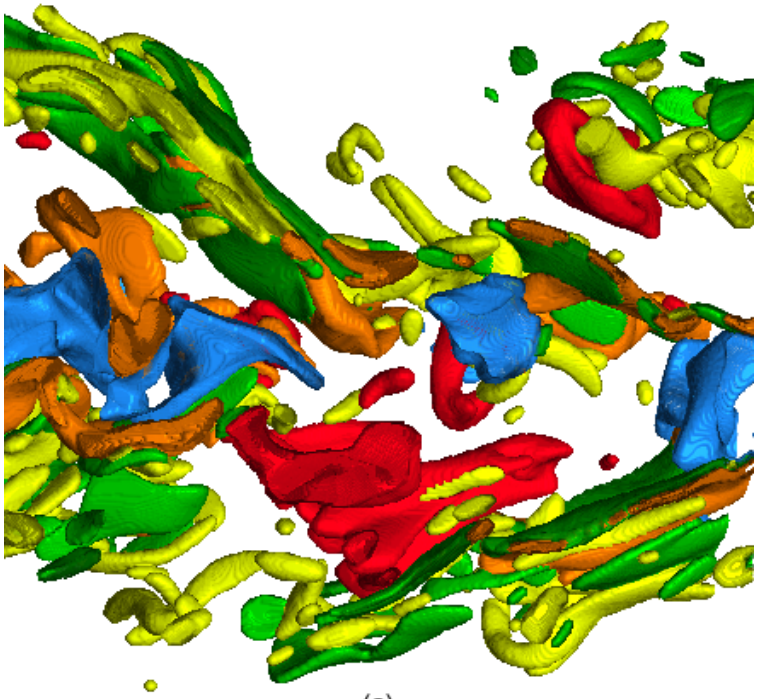


(a)

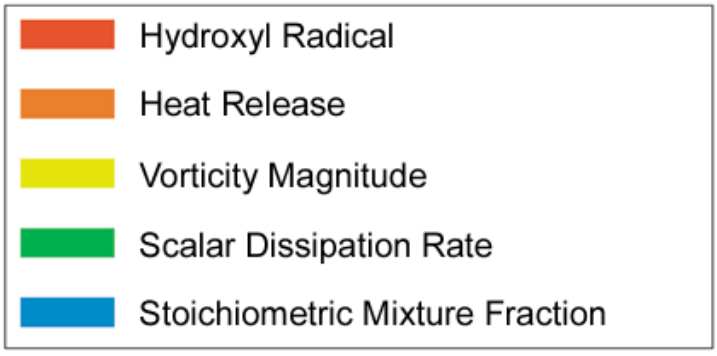


(b)

Figure 3.5: Rendering of the jet combustion dataset with the attributes' average values (a) and attributes' maximum values (b) used for the attribute target values. Attribute color key included in Figure 3.6

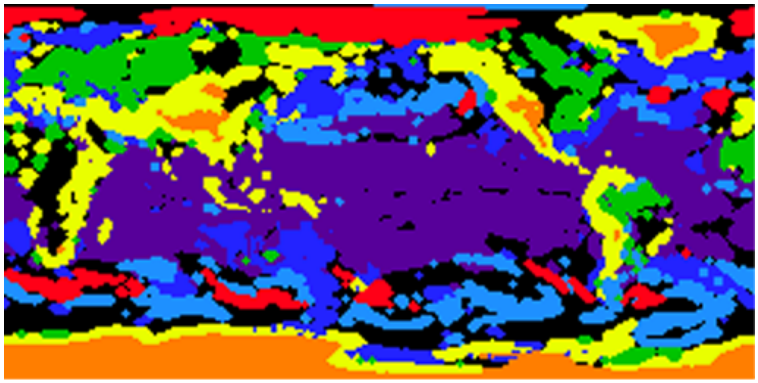


(a)



(b)

Figure 3.6: (a)Rendering of the jet combustion dataset with attributes' maximum values used for attribute target values, and all but 4.7% of points thresholded. (b) Attribute coloring key for combustion dataset images.



(a)

(b)

(c)

Figure 3.7: Climate dataset with maximum attribute target values. Default (a) and full (b) thresholding. (c) Attribute coloring key.

patterns. We can see the height of ice cap on Antarctica delineated by the abrupt ”jump” in surface geopotential. The same can be seen around Greenland. In other locations around the globe, only geographic regions such as Tibet, American Rockies and Andes show such abnormal surface geopotential due to their great altitudes.

3.3.3 Narrowing Exploration Space

Lastly, we give an example of how massSIV can be used for dataset exploration. In Figure 3.8 we show the output of our method run on a CO_2 measurement climate dataset over ten timesteps. This is a single variable dataset, so each color represents the values at a different timestep. Our program outputs the size of each attribute space, which suggests which timesteps contain the most impactful points. We also show the gray-scale image corresponding to the largest attribute subspace (red). This information can be used to determine which timestep a user should render, via a favorite single variable renderer.

3.4 Considerations of the Approach

To provide single summarizing images/volumes we employ a data reduction in attribute space. As with any approach utilizing simplification or compression, there are many concerns as to whether or not the results are oversimplified. To this end, our intention is for massSIV to be used in an application driven manner, so that the simplification can aid users in excluding areas not of interest.

Another issue is the sensitivity of target values. In our admittedly limited range of tested target values, we had few problems with sensitivity. However, there are cases in which this could be an issue, the following discussion can be used as a guide for appropriate uses of our approach. In cases of simple attributes, like the climate dataset's land ocean transition mask (1 for land, 0 for ocean), even changes in other attributes' target values could drastically alter its representing attribute subspace.

There are other cases of target value sensitivity, but they are relatively predictable. For instance, an attribute with a large variance would certainly differ (possibly greatly) with changes in target values. The inverse of this is shown in Figure 3.5, stoichiometric mixture fraction only slightly changes between its average and maximum value. Also, although we don't provide renderings

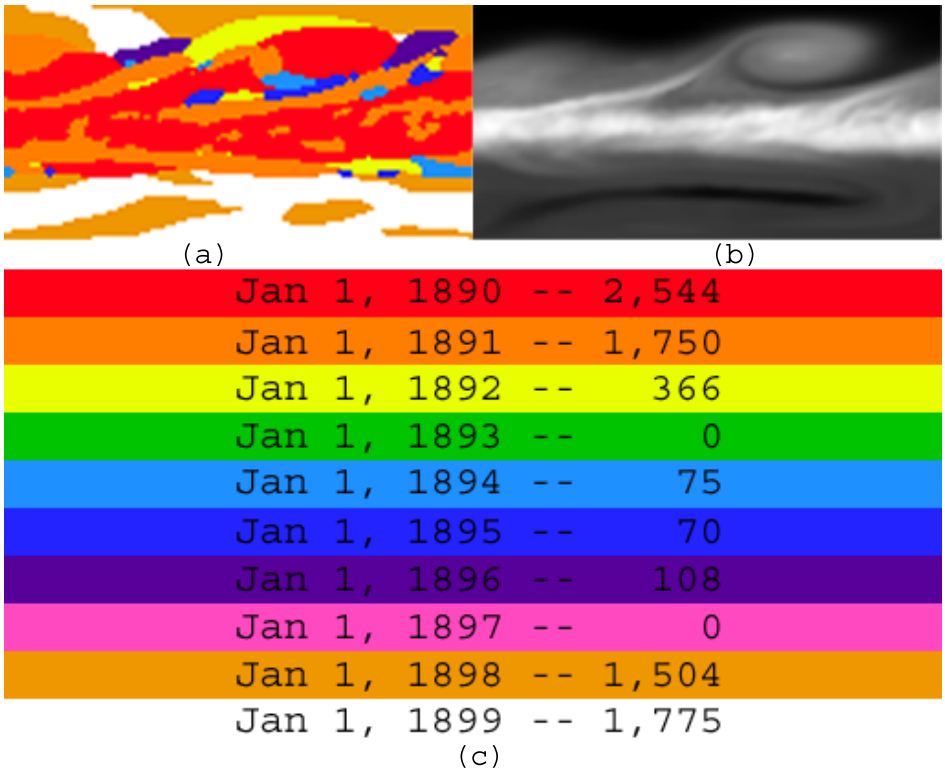


Figure 3.8: Concurrent rendering of 10 timesteps of the CO_2 measurement dataset (a). A sample gray-scale image of Jan. 1, 1890, the highest ranked attribute (b). The rankings of all 10 timesteps (c).

based on attribute relationships, attribute relationships could cause target value sensitivity. Areas where attributes are closely related represent possible areas where target values would be sensitive. However, this should only be an issue if those attributes are proportional throughout the volume.

Chapter 4

Application Study

Petascale computing has brought forth a transformational way of science. To the global effort on studying climate change, the transform has enabled not only tools more functional and more powerful than before, but also a more comprehensive scientific exploration than before. In this work, we report our efforts to employ recent ultrascale visualization technologies (SciDAC Ultravis) for studying terrestrial biogeochemistry datasets produced by computation (SciDAC C-LAMP) as well as surveyed by satellites. While many of the current efforts are specific to climate modeling research, the tools are generally applicable to other fields of computational sciences.

The Carbon-Land Model Intercomparison Project (C-LAMP), www.climatemodeling.org/c-lamp [Hoffman et al., 2007] was initiated to allow the international scientific community to thoroughly test and compare such terrestrial biogeochemistry models through a set of carefully crafted experiments. Well-defined metrics have been established for comparison of model results against best-available observational datasets, and models are graded on their scientific performance with respect to these metrics. Visualization tools and diagnostics are particularly helpful in uncovering model biases and discovering ways for improving individual models. In this chapter, we use the visualization technique outlined in Chapter 3 to compare models of different runs.

4.1 C-LAMP

The purpose of this model-measurement inter comparison is to allow the international scientific community to evaluate the performance of biogeochemical models normally coupled to general

circulation models (GCMs) [Hoffman et al., 2007]. Terrestrial models are scored based on their performance as compared to best-available site, field, and satellite observations through a rigorous set of metrics. To this end, we encourage you to provide feedback on the experimental protocol, the metrics used to evaluate model performance, and the observational datasets available for use in the inter comparison. The C-LAMP project bridges the climate modeling community and the measurement community. This role of C-LAMP enables significant potential of model improvement and more comprehensive measurement campaigns.

The C-LAMP project conducts two types of experiments. In the first type of experiments, biogeochemical land surface models are forced with an improved NCEP/NCAR reanalysis climate dataset. In these offline runs, the objective is to examine the ability of the models to reproduce surface carbon and energy fluxes at multiple sites, and to examine the influence of climate variability, prescribed atmospheric CO2 concentrations, and land cover change on terrestrial carbon fluxes during the 20th century, and specifically during the period for which the reanalysis data are available (1948-2004).

In the second type of experiments, an active atmosphere model is used to couple energy flows between the atmosphere and the terrestrial biosphere. However, atmospheric CO2 follows a prescribed trajectory for both steady state and transient components of the experiment. The prescribed CO2 is radioactively active and sea surface temperatures (SSTs) and ocean carbon fluxes are prescribed. The objective of these simulations is to examine the effect of a coupled biosphere-atmosphere on carbon fluxes and climate during the 20th century.

4.2 Parallel Query-Driven Visualization

Decadal to century time scale climate simulations typically output a large number of two- and three-dimensional variables at regular intervals, usually monthly. While identifying features is difficult, it is actually intuitive and practical to select only a subset of the data from within that high-dimensional variable space, to obtain a qualitative understanding of the overall results. This approach can be easily implemented if the resulting dataset can be stored entirely in-core. However, to handle larger datasets, a more sophisticated solution is necessary.

The solution we use is presented in [Glatter et al., 2006]. This approach involves designing specialized scalable visualization data servers with large-scale parallelism. Our index system in-

dexes general data items, including vertices, voxels, or particles, and is independent of grid type. The core data structure for indexing is an optimized M-ary search tree. The tree structure only amounts to approximately 1% the size of the whole dataset. The dataset can be stored externally on hard drives in a compressed manner. Only parts of the data that are used by the scientist are decompressed (and cached). The compression rates vary from dataset to dataset. For some typical datasets, we obtained a 20x compression rates, while we could obtain as low as a 4x compression on highly turbulent or noisy datasets. Using these rates, we conservatively estimate that a mid-sized cluster could already support parallel visualization of a dynamically queried dataset of 1 TeraByte (TB) size.

The M-ary tree uses a large branching factor, and serves the role of metadata to guide a search process. The branching factor is one of the primary differences between this search data structure and previous data structures, such as interval tree, k-d tree, quadtree and octree. Due to the large branching factor, M, the M-ary search tree requires little storage space. The data is not stored in the tree, but in a linear list sorted by a key function. The leaf nodes of the tree store only pointers to the respective data records in a sorted linear list. We have discovered that conventional methods to access records by traversing the tree are too expensive, both in terms of caching performance and the large number of addressing operations. We use a novel method to accelerate range searches in an M-ary tree to address this, optimized specifically for multivariate datasets [Glatter et al., 2006].

Data items are partitioned into groups by round-robin assignment according to high-dimensional space filling curve [Pascucci and Frank, 2001] order in attribute space. We use this type of data partition to distribute data amongst all visualization data servers equally. Thereby, we are able to achieve a nearly optimal load-balance for almost all kinds of queries. The M-ary search tree is then used to manage the data on each server. Our approach is relatively easy to deploy on networked commodity computers, whether clustered or not. The necessary number of parallel data servers depends on the size of the dataset.

4.3 Results and Discussion

The concurrent visualization method based on attribute subspaces summarizes multiple patterns according to their relative strengths so that overall trends can be visually compared, analyzed and comprehended.

4.3.1 The Models

As typical with any climate modeling efforts, the same models would be run under different conditions. To demonstrate our approach and typical use cases, we show three different types of runs: (i) control run (C-LAMP 1.2), (ii) varying climate transient run (C-LAMP 1.3) and (iii) varying climate, CO2 and N deposition transient run (C-LAMP 1.3). Under each type of run, there are two models from NCAR Community Climate System Model Version 3 (CCSM3), specifically CCSM3-CASA and CCSM3-CN. Hence we have in total 6 different simulation scenarios to visualize. The time span considered is the decade from 1990 till 1999. We show representative models of three individual variables: (2 meter air) temperature (TSA), rain and total soil liquid (TSL) that have been shown to be differentiating factors of climate regimes [Hoffman et al., 2005] in Figures 4.1 and 4.2. These figures are included to demonstrate such trends, i.e. the movement of temperature between low and high across months, or high/low amounts of rain in different geographical areas (precipitation in Antarctica is not in the form of rain, but snow instead). In Figure 4.3, in addition to these 3 variables we also concurrently examine 3 other variables: ground temperature, soil ice, and vegetation temperature.

4.3.2 Climate Patterns

In this section, we show major patterns of three variables: net ecosystem exchange of carbon (NEE), net primary production (NPP) and total leaf area index (TLAI). While we find that, in reference to the general climate trends of these three variables, the differences among models are visually discernible in several cases, within both models the results from 1.2 and 1.3 runs seem visually indistinguishable. For this reason, we do not show results from the 1.3 run. When viewing more than one variable in one setting, differences in models become apparent. In Figure 4.4 , high, low and mid mean that the target values for all three variables are chosen to be global maximum, global minimum and mid-way between global extremes, respectively. In terms of relative strength (or relative variation patterns) among NEE, NPP and TLAI, CASA and CN largely agree with high and low distributions in both January and July in 1990 (peak summer and peak winter months). The mid range distributions do reveal differences in high latitude areas, large deserts and mountainous (high elevation) regions.

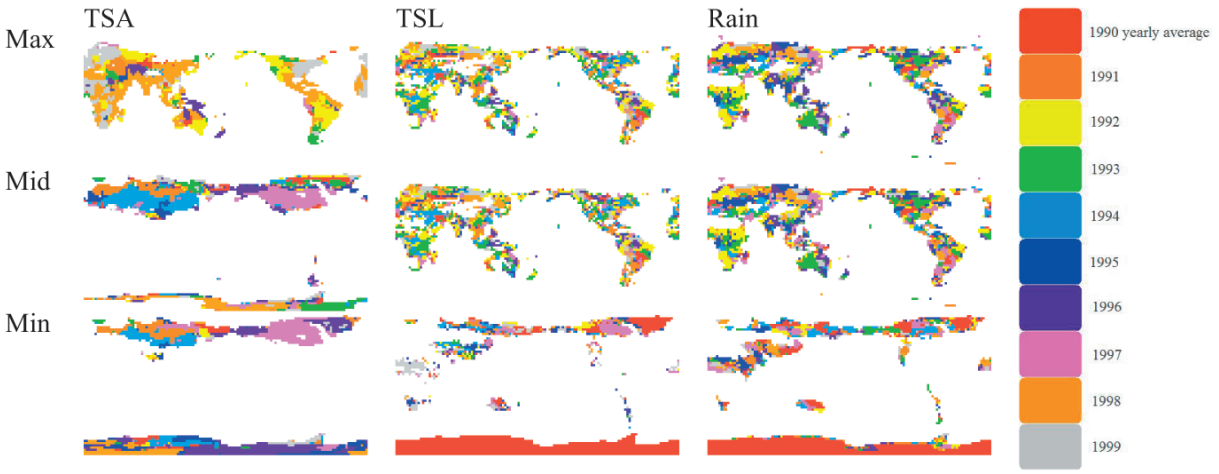


Figure 4.1: Year long average, 10 years in a decade. Which year is the closest to the maximum value (Max), mid-point of the possible range (Mid) and the minimum value (Min)?

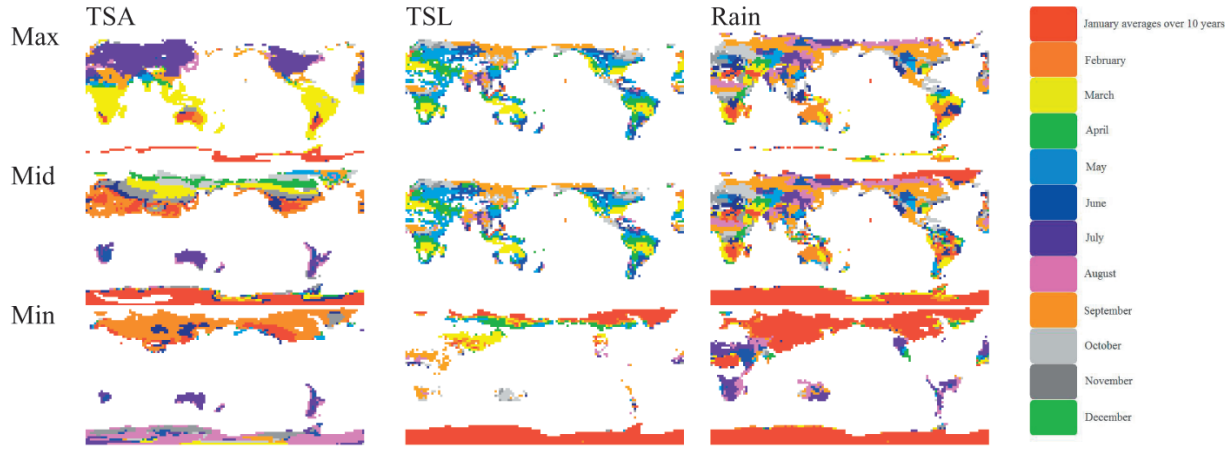


Figure 4.2: Per month average over 10 years. Which month is the closest to the maximum value (Max), mid-point of the possible range (Mid) and the minimum value (Min)?

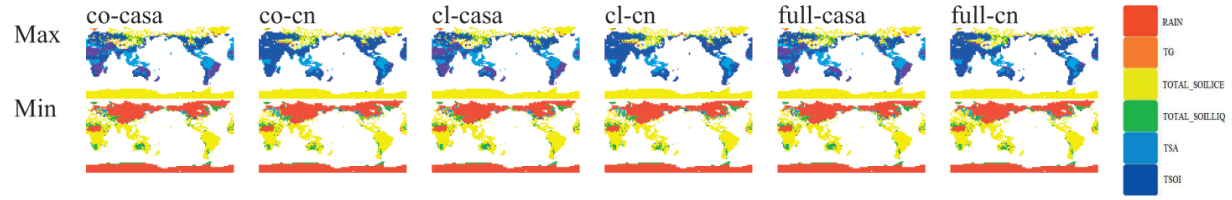


Figure 4.3: Showing 7 concurrent variables. For instance, in February 1990, results are very similar among all 6 scenarios, different only with which variable shows extreme high values.

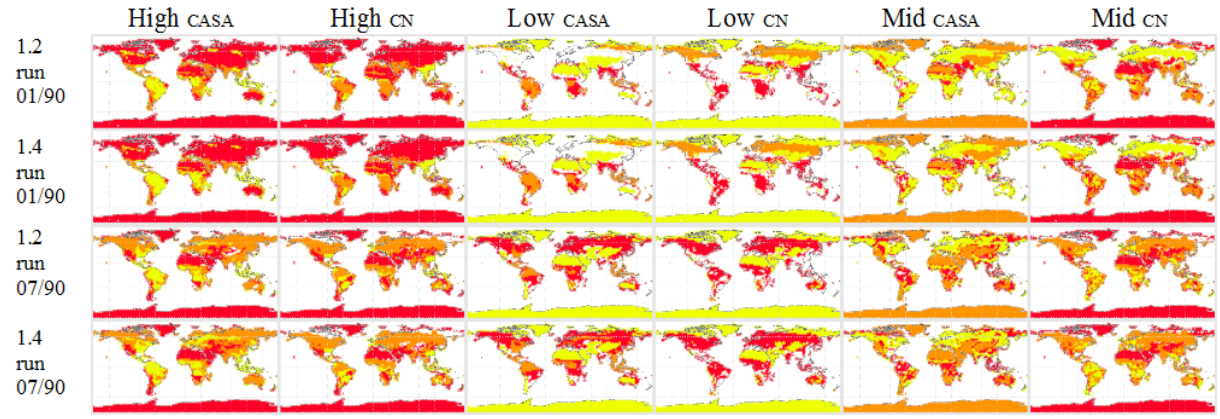


Figure 4.4: A location-specific summarizing visualization of extreme and normal relative distribution patterns among CASA and CN models in C-LAMP 1.2 and 1.4 runs. The variables considered are NEE (red color), NPP (orange color) and TLAI (yellow color) in January and July of 1990.

In Figure 4.5 with TLAI, 1.2 and 1.4 are visually similar. The heavier coupling in 1.4 does not seem to change relative strength in temporal distribution of yearly averages in the decade. Both CASA and CN seem to agree that 1990 has the lowest TLAI in the decade and in the geographic regions of barren lands (Arctic, Antarctic, Sahara and Tibet). As to relatively high TLAI in the decade, CASA still favors 1990 (closer to decadal-global extremes) in South America and sub-Sahara Africa. Also the earlier three years have high TLAI in all other areas of the globe. This pattern of earlier years in 90's showing high relative TLAI is not in CN runs. With both NEE and NPP, CASA and CN largely agree, but are, however, off by a year or two (both for high and lows) in some cases. For the NPP modeling results (both High-CASA and Low-CASA), the Amazon mostly shows the first half of the 90s to have lower values. In CN, the Amazon shows high NPP more uniformly distributed throughout the decade. No year during the 90s showed "abnormally" low NPP in CN models.

As another example, we calculate Pearsons correlation coefficient between pairs of variables for each simulation scenario over 1990-1999. By viewing all models concurrently, some model trends are clearly visible. Since 1.2 and 1.3 are so similar, only 1.2 is represented in Figure 4.6, i.e. no visible green or yellow. C-LAMP 1.4 is represented as blue and light blue (CASA and CN), so blue in an image represents the more coupled the run. NEE-NPP shows stronger positive correlation in more coupled runs in Eurasia, whereas NEE-TLAI shows stronger positive correlation more in less coupled runs in the same area. With respect to correlation we would expect this viewing to appear completely random, but the presence of contiguous areas of identical color highlight geographical regions of interest with a model.

In the future, we plan to engage in even closer collaboration with C-LAMP researchers because few of the model differences we have identified so far have clear explanations from the modeling community. We also plan more research for evaluating model differences in the context of model biases from real observational data.

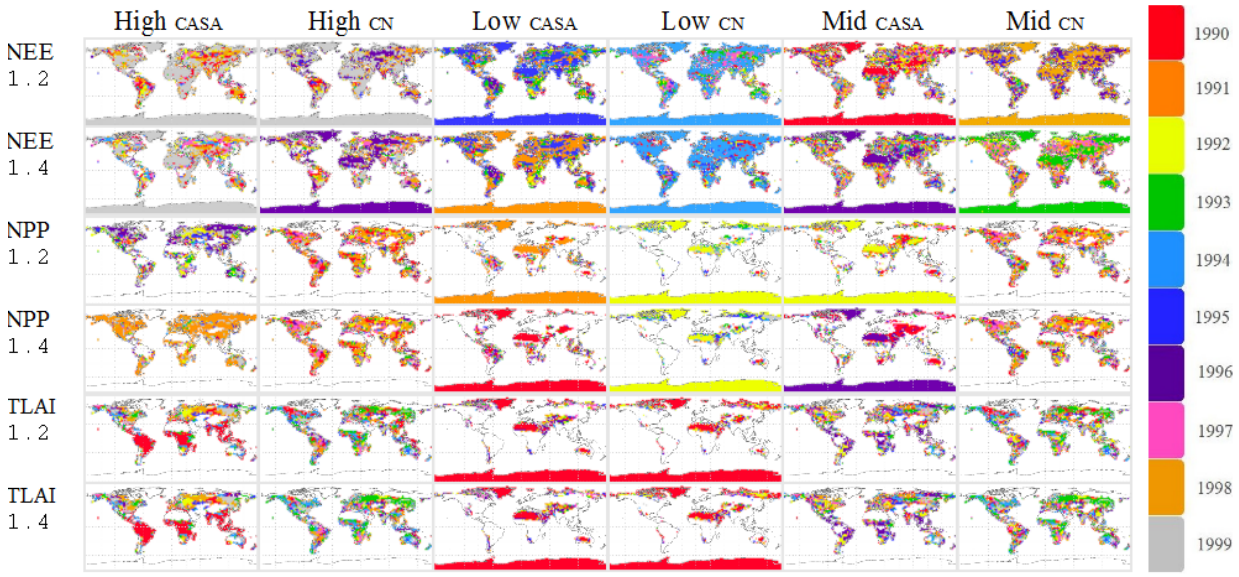


Figure 4.5: Year long average computed over 1990-1999. For each of NEE, NPP and TLAI and in 1.2 and 1.4 runs, which year is the closest to the decadal maximum (High), mid-point of the possible range (Mid) and the decadal minimum value (Low)?

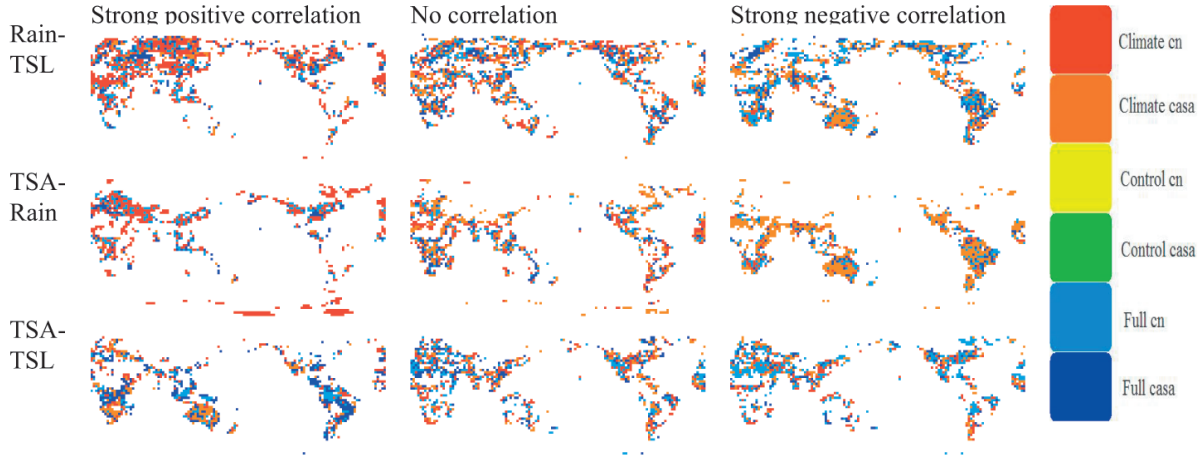


Figure 4.6: Model bias in bi-variate Pearsons correlation coefficient.

Chapter 5

Extending Attribute Subspaces

Today's scientific simulation at terascale and beyond has offered grand opportunities of innovative visualization research. Often mentioned in our discussions with domain scientists is the need to effectively study how multiple variables interact. In other words, this is to study the relationships among multiple variables in the simulation. The conceptual space most intuitive for the scientists to interpret such relationships is the value space formed by the variables. In this chapter, we use the term "relationship" to specifically mean value space relationships.

The de facto tool used by application scientists to visualize relationships is the scatter plot. To a certain degree, parallel coordinates as a variation of scatter plots are often considered as well. However, these tools are mainly dedicated to studying one (or more) bivariate relationships and do not scale up well to handle more concurrent variables. In addition, it is also quite hard to visually examine many scatter plots together even if the plots are placed side-by-side.

When multiple spatial and temporal domains need to be studied simultaneously, more sophisticated solutions are called for. This is especially necessary for high-resolution time-varying datasets with a large number of variables. Current tools are particularly hard to use if a user desires to study multivariate relationships in a spatially or temporally specific way. We address these issues specifically by providing a data-centric categorical visualization that is still in the original spatial context of the data.

In this work we present a general framework for concurrent visualization of unconstrained multivariate relationships via relationship specification. Although there have been advances in visualizing multifield data with complicated relational objects, such as glyphs or textures, we

have decided to take a step back and view the original problem associated with understanding multivariate data. For this reason, we simply use color as the differentiating factor, and in fact, human color perception is the practical limit on the number of multivariate relationships that can be simultaneously considered. Even in this setting, we are able to show significant results.

Our approach is to allow for the specification of several multivariate relationships, and then to classify each point as belonging to one of these relationships. A possible interpretation is that this is a competition process, and each point is assigned the "winning" relationship, i.e. the best fitting. Specifically, we specify a relationship as a set of n-dimensional points, for a relationship among n variables in the dataset. We can show a single view of the data that summarizes the existence of all specified relationships. We can fuse key aspects of both multiple attributes of the data and prominent inter-variable relationships. Such images not only provide a user the capability to confirm suspected relationships, but also allows for the discovery of new ones. Also, we do not exclude the viewing of univariate features, these are still an option for specification, since we treat univariate relationships as a special case of multivariate relationships.

We designed a relationship specification framework to discretely specify multivariate relationships in our framework. In a way, one can conceptualize our relationship specification as picking discrete dots in the high dimensional value space. The visualization software stays strictly independent of how those "dots" are chosen, either manually by a user or automatically generated at high resolutions by a program. There are no limits in the visualization software as to how many dots can be included in each specified relationship, nor is there a limit on how many relationships can be handled. This approach is versatile and powerful in that the classes of relationships that can be specified are not restricted, and could be customized to match domain specialties.

5.1 Multivariate Relationships

In a general sense, we take a programming language approach to allowing for the specification of unconstrained specification of relationships. For this approach to be effective, each level in the heirarchy from user interface to point classification must operate in a space that allows for specifications with a balance of intuition and robustness. In the remainder of this section, we discuss the concepts of two of these levels, the relationship space the user interacts with, and how the points are classified based on this interaction, i.e. our relationship competition process.

5.1.1 Relationship Space

A concept central to our approach is the notion of unconstrained multivariate relationships. We provide single renderings that are simultaneously based on all user specified relationships. We intend for our approach to aid not only users attempting to learn new information from a dataset, but also experts of the dataset to extract complex suspected features. For this reason, we treat relationships as generally as possible. We consider a typical relationship between n variables, $v_1, v_2, \ldots, v_n$, to exist in the discrete space $v_1 \times v_2 \times \cdots \times v_n \in \mathbb{R}^n$. However, we allow the specification of a relationship to be any union of sets of points existing in this space, whether they be an arbitrary set of points, or a set of points logically equivalent to a function with domain and range in $\mathbb{R}^n$. Figure 5.1 is an example of a logical, and legal, specification of three two-variable relationships, along with typical specified relationships used in our renderings. For ease of discussion, in the remainder of this section, discussions are in terms of bivariate relationships.

5.1.2 Competition Between Relationships

Relationships are provided as user input to our system as a set of 2-tuples specifying the relationship ($n-$tuples for a relationship between n variables). Each tuple may be interpreted as: "show us everywhere variable one is near coordinate 1 in the tuple, and variable two is near coordinate 2." For example, for a two variable and normalized dataset, if we are interested in regions where both variables are at a minimum or where both variables are at a maximum, we would specify the relationship as: $[0, 0], [1, 1]$. We also allow for a coordinate in the tuple to contain more than one value, i.e. a relationship may specify that interesting areas are where a variable is at a specific value while the other is one of many values, e.g. we now want to view regions where variable 1 is at a maximum or minimum value and variable 2 is at a minimum: $[(0, 1), 0]$. This is exactly how the system defaults $vertical$ and $horizontal$ are defined, see Section 5.1.3. To provide a summary of all user specified relationships, we employ a best fit relationship competition. We first assign to each relationship its own color. Then, our rendering is basically a display at every point of the relationship, if any, that best fits at that location. However, there are several tests a relationship must pass to stay in the competition.

As stated above, a relationship is specified as a set of tuples, with each variable in the relationship represented in each tuple with either a single value or a set of values. In other words, for each

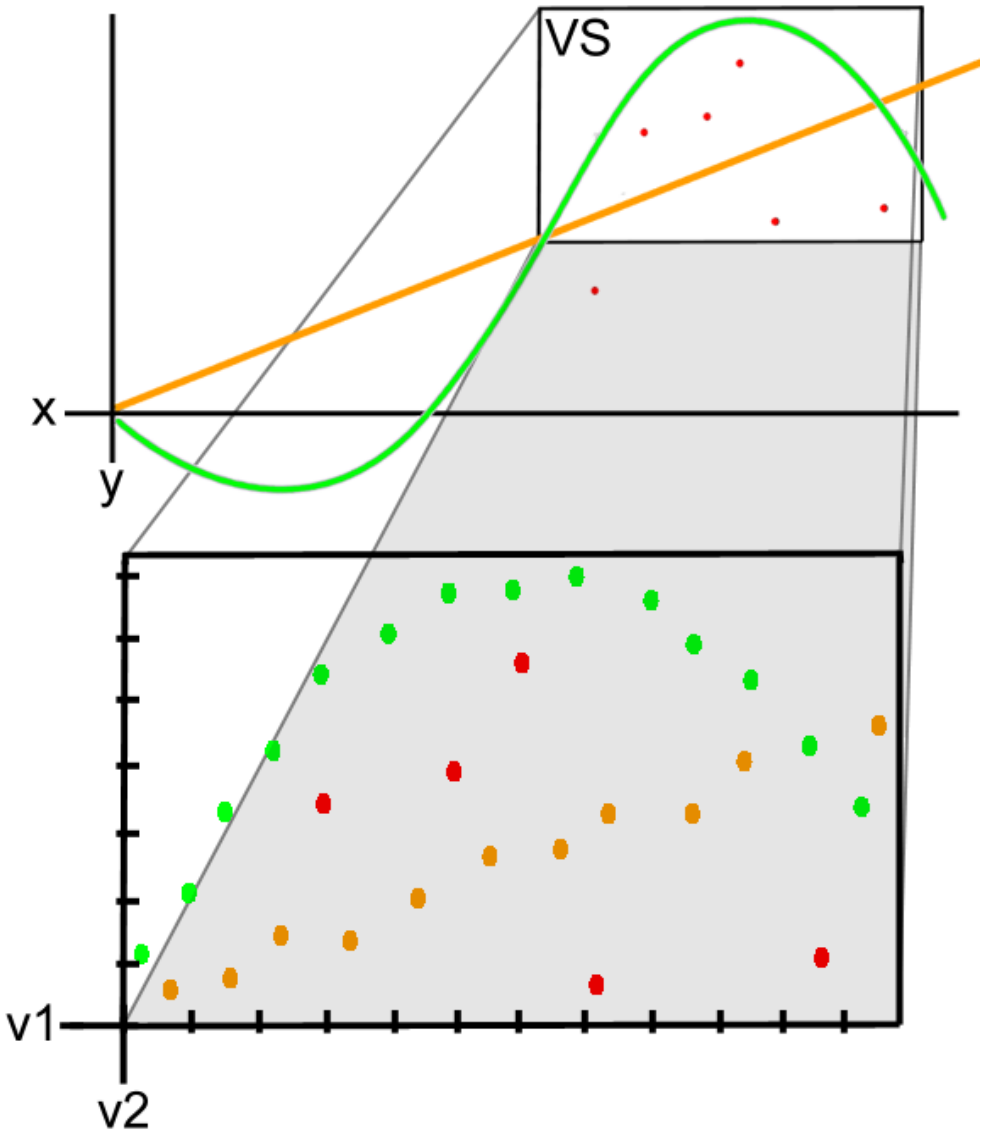


Figure 5.1: Three bivariate relationships specified in $\mathbb{R}^2$ and their corresponding analogues in discrete variable space (VS).

variable there is a set of values belonging to different tuples that specify its role in a relationship. For the first test, we ensure that the closest of these values for each variable to the location at hand, comes from the same tuple. For the following relationship: $[0, 0], [1, 1]$ at a spatial location with values: .25, .75 for variables 1 and 2, the values in the relationship specification that are nearest the location's values are 0 and 1 respectively, which belong to different tuples. Therefore, this relationship is no longer considered at this location. Second, a variable must be sufficiently close to the best fitting relationship. This is particularly important, since we calculate how well a relationship fits a point simply as the average distance between each variable involved and a location's value. The smaller this average distance is, the better a relationship fits that point. We normalize this distance, and refer to one minus this value as a relationship's score at a point.

Clearly, if one variable in a tuple is extremely close to that variable at the spatial location, and another is far away, the relationship is not a good fit for those two variables at that location and should not be considered, even though the score may be relatively high. We allow the user to specify how far away any one value of a tuple should be from the variable's value at a point and remain in the competition. As a default, we calculate the average distance for each variable in a relationship to all spatial locations, and enforce that each value in a relationship specification be at least this close. Lastly, we consider the relationship with the highest score at a point to be that point's winning relationship. The output of our method is then a tag volume, each voxel contains only an index indicating the winning relationship.

5.1.3 Default Settings in Our System

In the field of multivariate relationship visualization, the most researched and well known types of relationships are ones that are two-variable that exist in the space $v_1 \times v_2$. These relationships are also the most intuitive. Although relationships may be specified among any number of variables with our approach, we assume the usual use of our approach to be visualizing sets of two-variable relationships. For this reason, we provide eight two-variable relationships as defaults easily accessible in our system.

Figure 5.2 shows the logical representation of these relationships; many of the results in this work are generated from different combinations of these. It should be noted that logically continuous relationships are still encapsulated by this approach. We decided to keep the specification

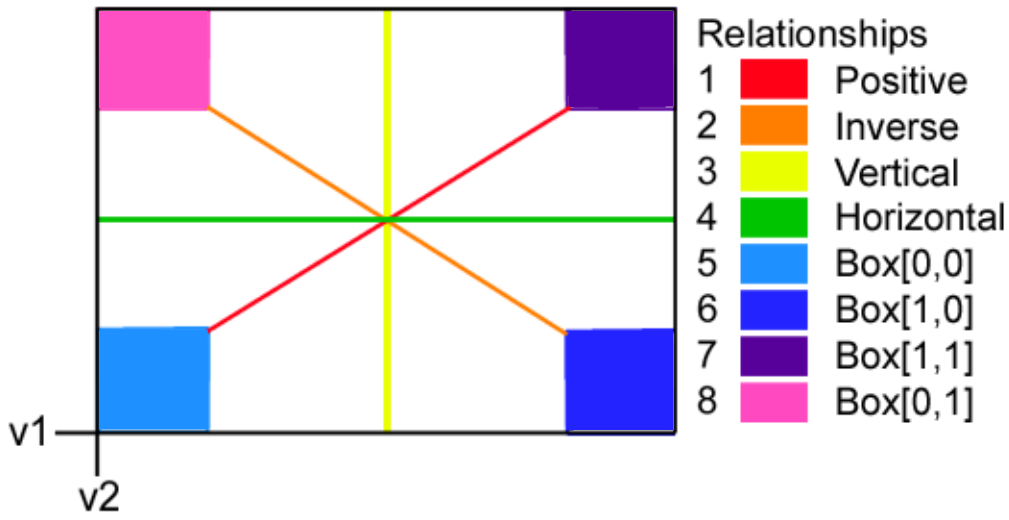


Figure 5.2: Default relationships offered in our system.

discrete to simply stay in the context of actual values of the data. So, the lines in Figure 5.2 would be a set of points. However, since we employ a best fit process to compare relationships, each point would actually be a box, and the result would be equivalent to the line seen, except it would have width. This width represents how loosely a relationship could fit a point and still vie for the best fit.

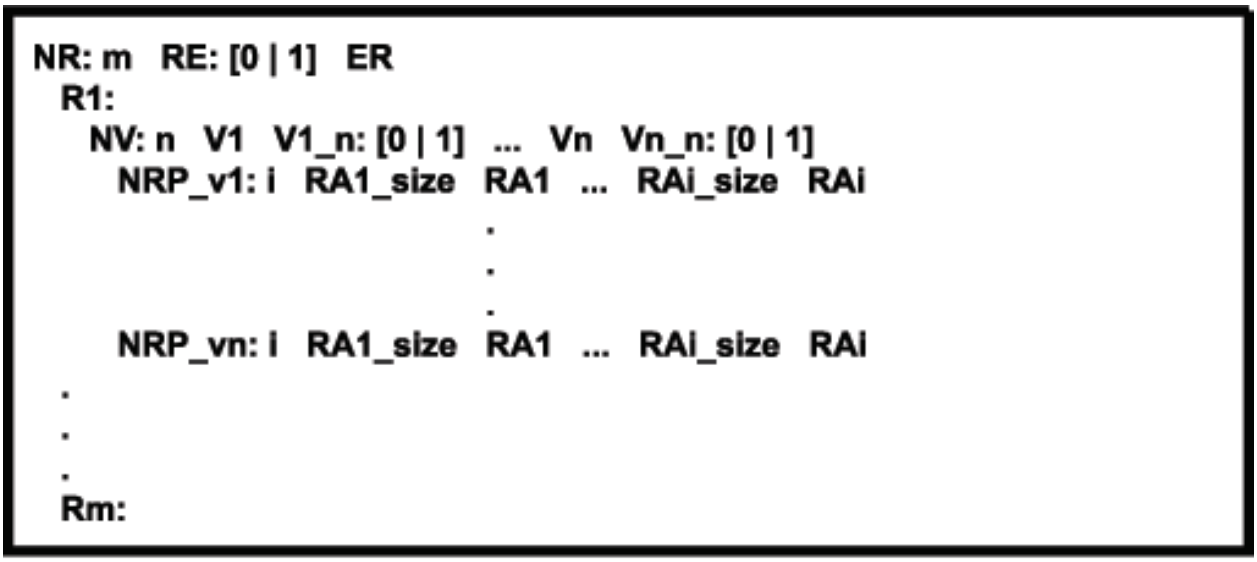
5.2 Relationship Specification

Our system performs its relationship competition calculation based on some set of user specified relationships. We attempt to maintain relationship flexibility, which requires a very specific and robust template for relationship specification. In this section we introduce our framework for the user's specification of relationships: the RS-file.

5.2.1 RS-Files and Their Uses

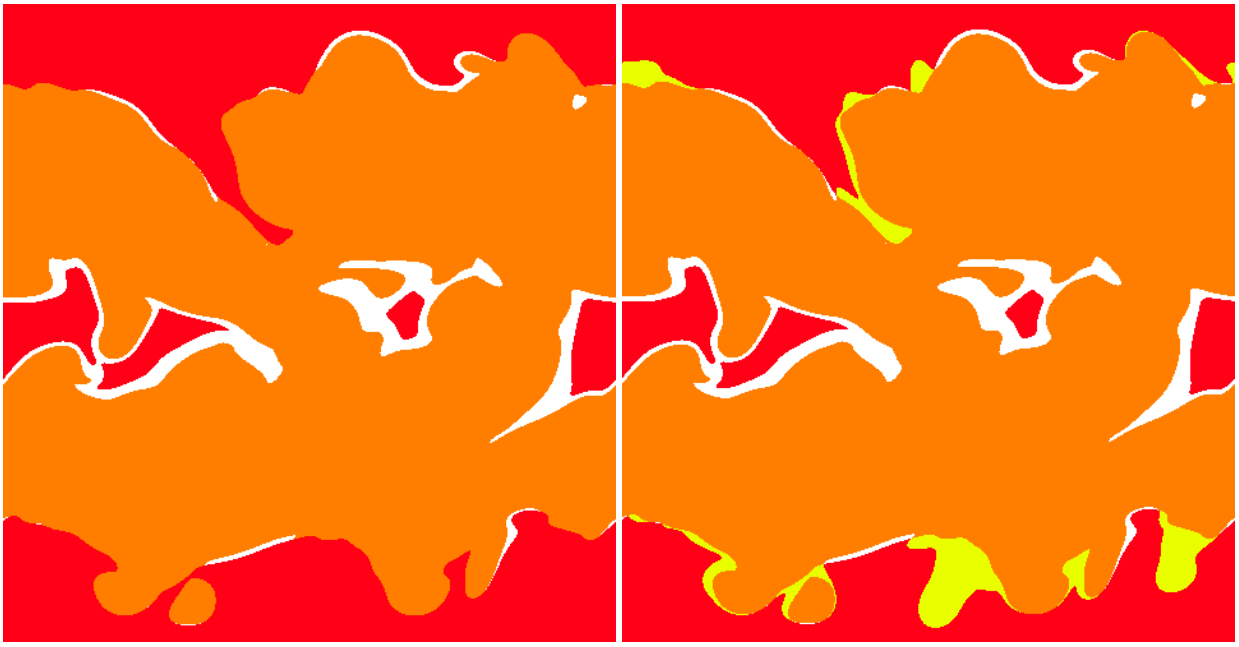
Figure 5.3 depicts the framework for an RS-file. We now describe in detail the more subtle aspects of these specification files. The first fields of note are the relationship exaggeration and the exaggeration rate. Since relationships may be specified univariately, there are some cases where this value is necessary. For instance, in the combustion dataset, there is a well known area where OH is near 0 and mix-frac is near .42. If we specify a relationship file to view this relationship but also to view the univariate areas near 0 and .42, for OH and mix-frac respectively, there are no areas where the two-variable relationship is a better fit than both univariate ones. Figure 5.4 shows the benefit of using the relationship exaggeration rate to raise the score of the relationship over those of the univariate ones.

In Figure 5.5 we show how relationships may be combined in an effort to maximize information contained in our renderings. In this figure, we show two simple relationships: the first, shown in red, is where OH and heat release are near their maximum values, the orange is where the same two variables are near their minimums. In Figure 5.5 (b) these relationships are combined into a single relationship, this image is the output of our method run on RS-file (d). This kind of operation is particularly useful for starting with simple relationships and building more complex and revealing images. Also, we allow for a univariate relationship to be merged with another relationship by using the wildcard, $\star$. This allows the user to specify that a two-variable



NR	- Number of relationships, m in this case.
RE	- Exaggerate relationships? [0 | 1]
ER	- Rate of exaggeration
R*	- Relationship specifications 1 through m
NV	- Number of variables in this relationship, n for R1.
V*	- Variables considered in this relationship.
V*_n	- Are the variables' relationship points in the range [0,1]? [0 | 1]
NRP_v*	- Number of relationship ranges per variable.
RA*_size	- Size of each range.
RA*	- Range specifications.

Figure 5.3: RS-file format.



(a) (b)

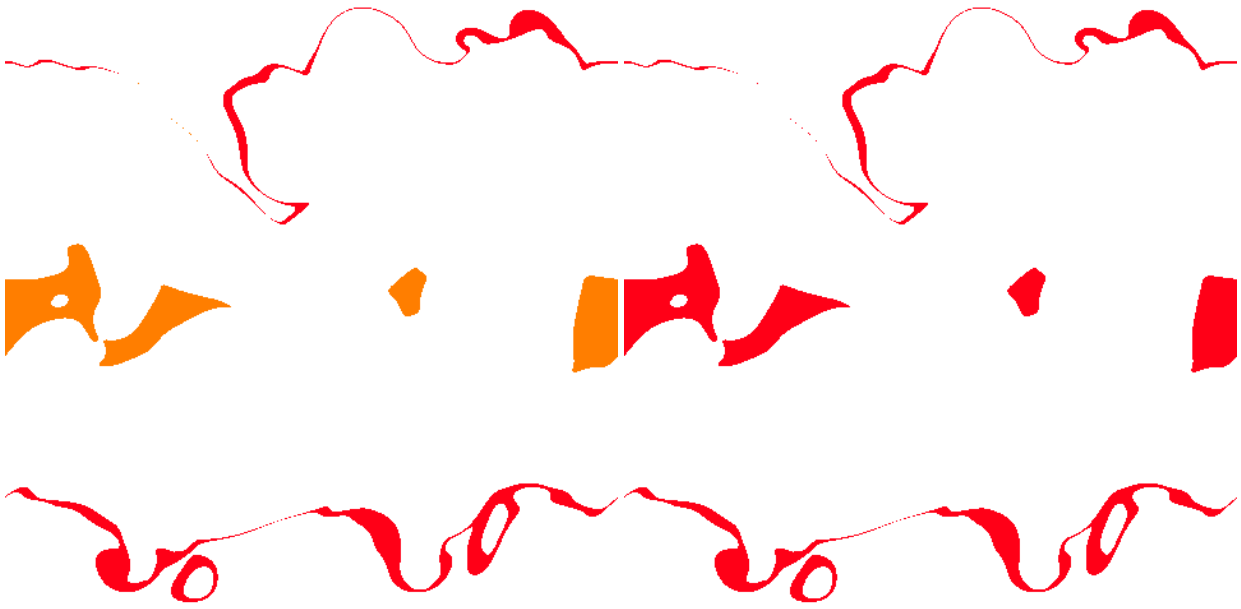
```
NR: 3  RE: 0
  R1:
    NV: 1  V1: 1  V1_n: 1
      NRP_v1: 1  RA1_size:1  RA1: 0
  R2:
    NV: 1  V1: 5  V1_n: 0
      NRP_v1: 1  RA1_size:1  RA1: .42
  R3:
    NV: 2  V1: 1  V1_n: 1  V2: 5  V2_n: 0
      NRP_v1: 1  RA1_size:1  RA1: 0
      NRP_v2: 1  RA1_size:1  RA1: .42
```

(c)

```
NR: 3  RE: 1  ER: 1.25
  R1:
    NV: 1  V1: 1  V1_n: 1
      NRP_v1: 1  RA1_size:1  RA1: 0
  R2:
    NV: 1  V1: 5  V1_n: 0
      NRP_v1: 1  RA1_size:1  RA1: .42
  R3:
    NV: 2  V1: 1  V1_n: 1  V2: 5  V2_n: 0
      NRP_v1: 1  RA1_size:1  RA1: 0
      NRP_v2: 1  RA1_size:1  RA1: .42
```

(d)

Figure 5.4: The red and orange in (a) and (b) correspond to OH near 0 and mix-frac near .42 respectively. (b) Image showing the relationship where OH is near 0 and mix-frac is near .42 (yellow). (c) The relationship specification file for the first image. (d) The relationship specification file for the second, the only difference being the exaggeration rate.



(a) (b)

NR: 2 RE: 0
R1:
NV: 2 V1: 1 V1_n: 1 V2: 2 V2_n: 1
NRP_v1: 1 RA1_size:1 RA1:1
NRP_v2: 1 RA1_size:1 RA1:1
R2:
NV: 2 V1: 1 V1_n: 1 V2: 2 V2_n: 1
NRP_v1: 1 RA1_size:1 RA1:0
NRP_v2: 1 RA1_size:1 RA1:0

(c)

NR: 1 RE: 0
R1:
NV: 2 V1: 1 V1_n: 1 V2: 2 V2_n: 1
N_v1: 1 R_s:1 RA1: 0 R_s: 1 RA2: 1
N_v2: 1 R_s:1 RA1: 0 R_s: 1 RA2: 1

(d)

Figure 5.5: Example of relationship combination.

relationship is a good match at a point if only one of the variables is close to some special value.

In the two previous examples, notice all variables' possible relationship points except for mix-frac's are given in the range [0,1]. We do this because we normalize the dataset before any calculations take place. However, in the case that a specific value in the variable space is desired in the relationship specification rather than some general relationship, V_n may be set to 0 to signal our system to normalize the relationship point. Lastly, to use one of the system's defaults, the user needs only to set NRP to 0 for the first variable, then specify the relationship number: 1 through 8.

5.2.2 Rendering with Scores

Since we color every point based on the relationship that best fits it, we realize that the winning relationship is not necessarily a great fit. For this reason, we take the score of the winning relationship at each point, and output a second volume containing these values. For the remainder of this chapter, we present images where the colors from winning relationships are composited with the scores of how well the relationship actually fits.

Figure 5.6 shows the coloring of winning relationships, the scores of these relationships, and the composite of the two. For better contrast, we show the inverse of the scores image.

5.3 Demonstration of System

In this section, we demonstrate our approach on three simulation datasets from different domains. The first is a global climate modeling simulation containing 86 variables simulated hourly from the year 2000 till 2099. The second dataset is the 128^3 vortex dataset, and our images come from timesteps 89-98. Lastly, we use a dataset from combustion research, with a grid size of $480 \times 720 \times 120$ and 5 variables. Our renderings come from timesteps 116 and 117.

Our intention for this approach is to work in the context of volumetric data. We have achieved some compelling results with simple testing. We mainly use small numbers of clearly distinguishable functional categories over a few bivariations for fundamentally spatial datasets that have a reasonable granularity of functional variation with respect to graphical output.

(a) (b)

(c)

Figure 5.6: (a) Combustion dataset colored by winning relationships: red is low OH and heat release, orange corresponds to high vorticity magnitude and scalar dissipation rate, yellow is low OH and mix-frac near .42. (b) Relationship Scores. (c) Composited image.

5.3.1 Climate

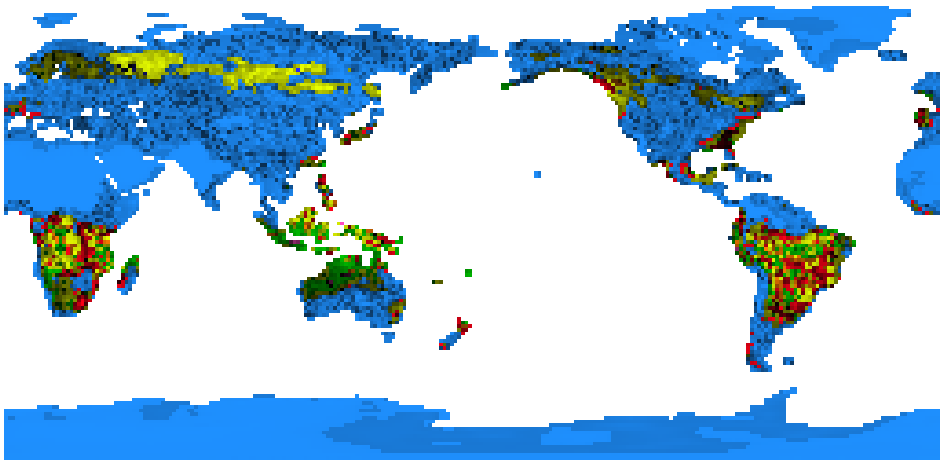
The climate dataset consists of 86 variables output from a global land surface model running within a general circulation model performing a projection of Earth's climate from the year 2000 to 2099. It has been shown this simulation result matches well with recorded climate observations. We show images created using the monthly averaged data from January and July 2000 to reveal patterns containing seasonal extremes.

In Figure 5.7 we show relationships specified for two conceptually related variables: intercepted water vs. atmospheric rain. Figure 5.7(a) shows the month of January in the year 2000, and Figure 5.7(b) shows the same visualization for July. Our results both confirm our common understanding and also reveal less commonly well known patterns.

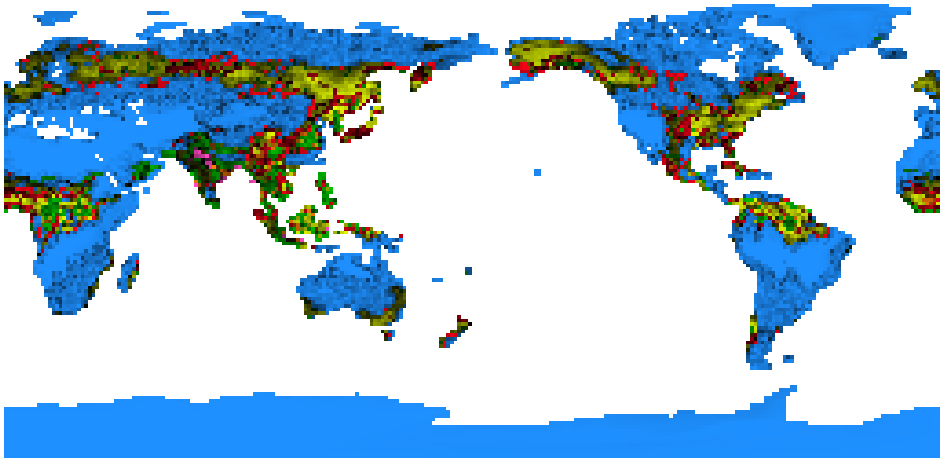
January and July are the peak months of winter and summer (Northern and Southern atmospheric patterns are just the opposite). Regardless of the season, desert areas show blue color (i.e. low precipitation in the form of rain as well as low intercepted water on the ground). Polar areas, such as Antarctica, Greenland, Northern Canada and Eurasia, have low rainfall during winter because most precipitation is in the form of winter storms. Due to low winter temperatures, intercepted water on ground is also almost non-existent. The fact that intercepted water does not significantly rise even during peak summer months indicates a still low level of glacier melting in 2000.

In other areas, for instance in the U.S., well known weather patterns of summer and winter are also apparent. During winter months, most inland areas of the U.S. show less significant rainfall, except the Pacific Northwest and coastal areas. Also in January, South America and sub-Sahara African areas are in the middle of wet season. At locations receiving significant rainfalls, the amount of intercepted water would still have to depend on geographic and vegetation features. In July, Southwest U.S. goes through a drought in 2000, while the other areas get significant rainfall from the usual summer weather patterns, such as thunderstorms or hurricane related precipitation. However, due to the usual low ground soil moisture levels and rich vegetation in the Southeast U.S., intercepted ground water is not typically at high levels.

The dark color pigmented pixels that appear as noise signify locations where the winning relationship is not really a strong one. There are also black pixels in the rendering. Those pixels represent locations where none of the default relationships are of good enough match (i.e. de-



(a)



(b)

Figure 5.7: Climate images created by using the eight two-variable relationships shown in Figure 5.2 (with the same color scheme). The two variables are intercepted water vs. atmospheric rain. (a) January 2000. (b) July 2000.

tectable by thresholding the per pixel score). We do not fully understand the cause of such effects without further in-depth understanding of the climate model. However, we do believe those dark pigmented pixels could suggest to climate scientists possible directions for further investigation.

Results such as those shown in Figure 5.7 are typical with our tool. However, although the climate dataset represents the most intuitive results we can generate, and we are capable of providing a user with a rendering of multiple many-variable relationships, there is still an issue of intuitiveness and interpretability. In Figure 5.8 we provide a rendering based on nine three-variable relationships. The three variables are: QDRIP (throughfall), QINFL (infiltration), and QINTR (interception). It is known by the climate community that these variables are generally $positive-$ly related, and indeed, the orange in Figure 5.8 corresponds to places where all variables are near their minimal values. However, the problem is in the other areas, it is difficult to understand the implications of relationships in high dimension, even if one knows of their existence. We believe our method will continue to provide useful results even as domain knowledge breaks the barrier of understanding high dimensional relationships.

5.3.2 Vortex

A key benefit to studying multivariate relationships is that by nature, important information about a dataset can be learned even if a relationship doesn't very well fit the data. The 128^3 vortex simulation dataset has been studied in-depth and is widely understood. The key features in this dataset are known to occur around variable values in the range of [5.0, 5.5], and exhibit a high amount of change between timesteps.

In Figure 5.9 (a), we show voxel values (i.e. vorticity) in the volume that are $positive-$ly related between two consecutive timesteps. This image is particularly interesting in that the only areas that are not colored by this relationship are the well known features of the vortex dataset. We show these features in Figure 5.9 (b), each of the nine colors correspond to areas of the volume where values in consecutive timesteps are near 5.25. This image then shows, the propagation of interesting features throughout time. Note, in this image, the white areas are where no relationships even remotely fit the data, so therefore nowhere near the range of [5.0, 5.5]. This area is also the area in the first image corresponding to $positive-$ly related timesteps. In those two images, we have shown the same areas, but each representing different aspects of the data. In fact, they

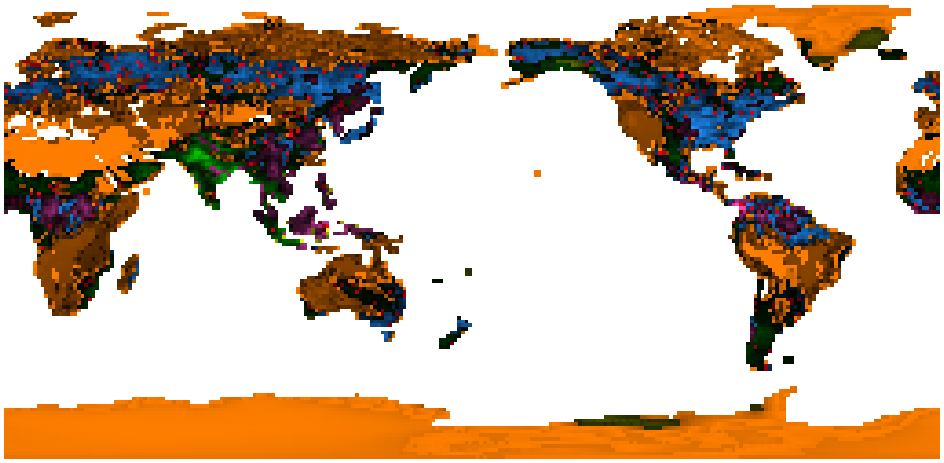
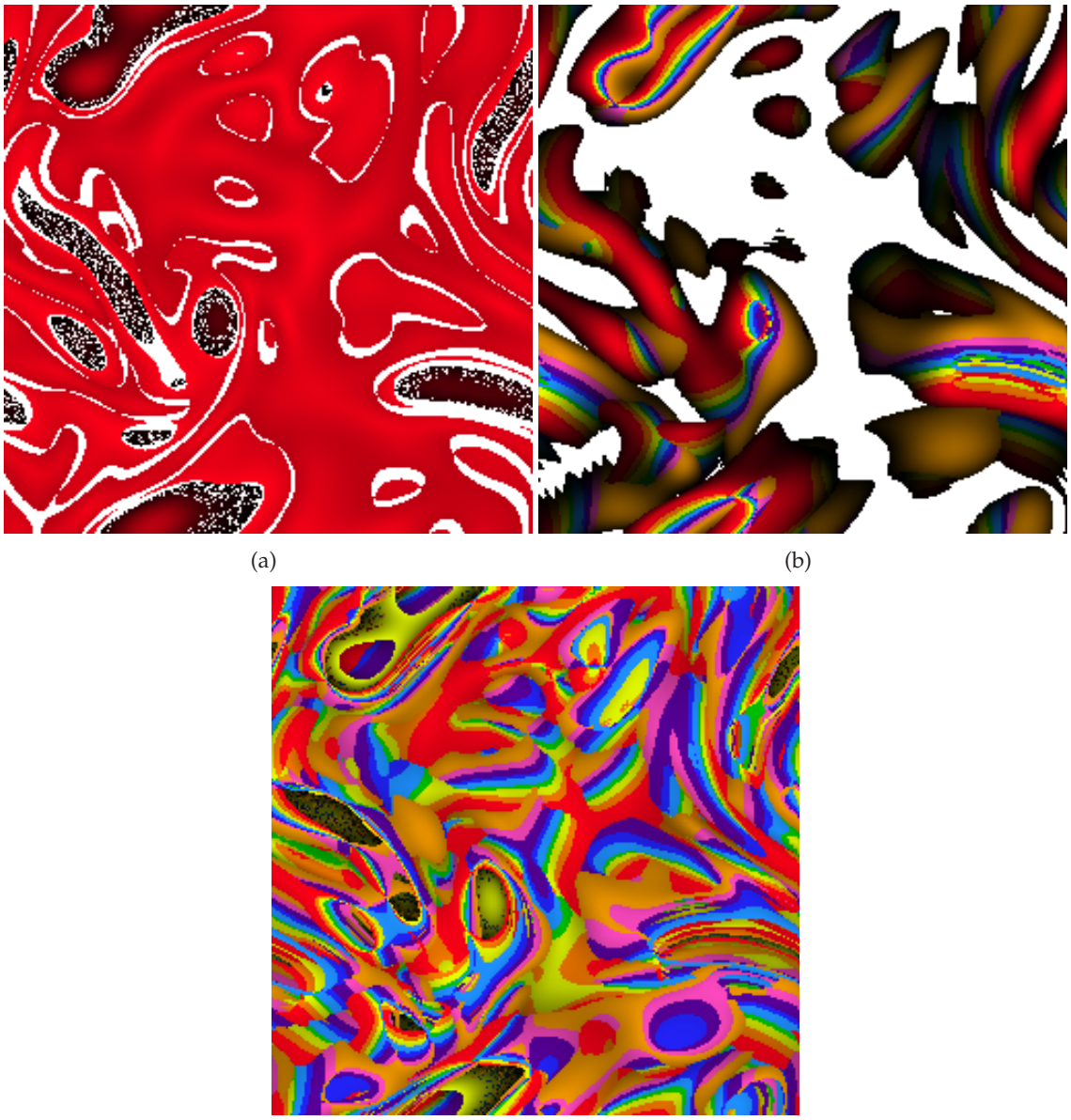


Figure 5.8: Climate image calculated from 9 3-variable relationships, month: July. These relationships correspond to the 3D representation of the $positive$ relationship and the eight 3D representations of the Box relationships from Figure 5.2.



(a) (b)

(c)

Figure 5.9: Sample images from the Vortex dataset, timesteps: 89-98.

are opposite images, even though the relationships are not. Finally, in Figure 5.9 (c), we provide an image similar to both of the previous images. In this image, we are showing areas where values in consecutive timesteps are $positive-$ly related.

This time, considering all 10 timesteps pair-wise from 89 to 98 in a sequential manner.

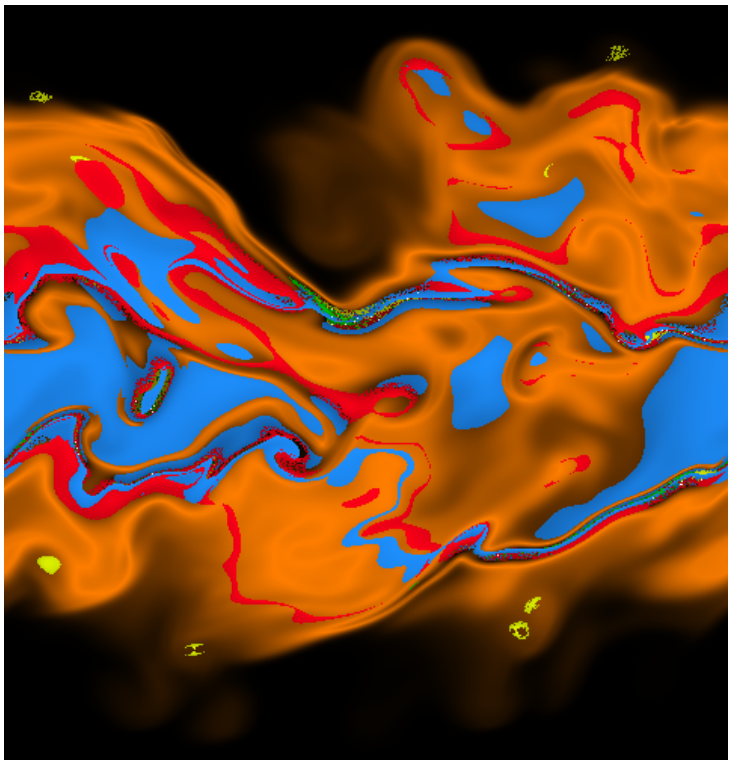
5.3.3 Combustion

The ability to quickly generate RS-files encourages users to view relationships that could potentially produce unexpected results. For the images we have created from the combustion dataset, we use all five variables over two timesteps, hence ten attributes. Figure 5.10(a) shows areas in which a variable is $positive-$ly related to itself over two timesteps. In this case each color still represents a relationship, but more specifically is applicable to single variables as well. In both Figure 5.10 (a) and (b) the color scheme represents relationships as follows: OH (red), heat release (orange), vorticity magnitude (yellow), scalar dissipation rate (green), mixture fraction (blue). Also, in both images there are areas with a grainy appearance. These areas represent places where multiple relationships' scores are very close to each other. Grainy places can therefore be used as a guide for narrowing down rendered relationships.

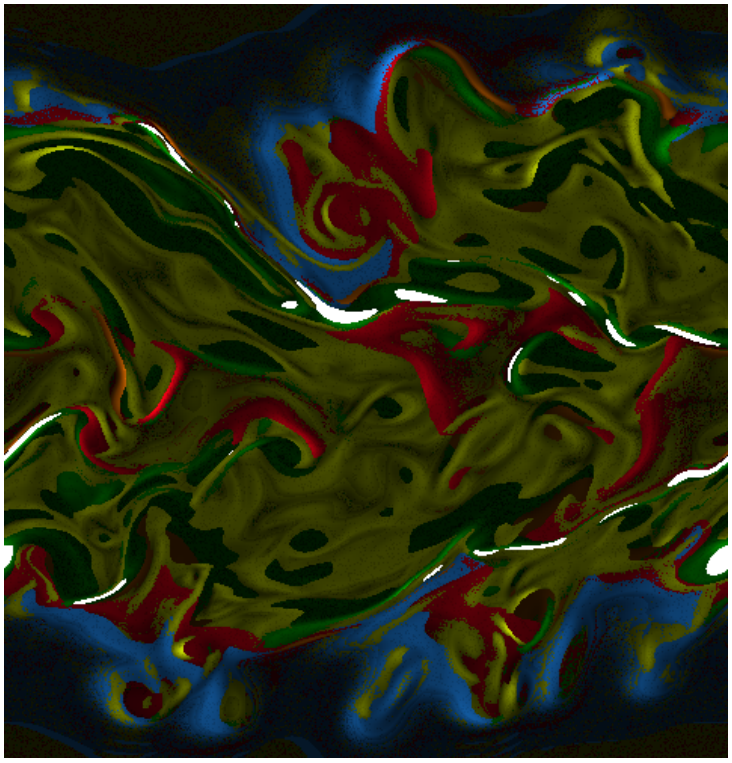
Figure 5.10 (b) is the opposite of (a). For this image we created an RS-file that was a request for areas where variables are $inverse-$ly related to themselves over two timesteps. Although in only few places are the relationship scores high, the clear presence of structure suggests that there is great potential in studying these unusual relationships. We feel that our approach is extremely well suited for the discovery of new features in data.

5.3.4 3D Renderings

Using the mechanism of competition to show multiple relationships is indeed just another classification method to volume visualization. The resulting volume contains on each voxel a tag (i.e. segmented) and a strength score (which can be treated as opacity). There are a number of volume visualization techniques that can be directly applied to render such volumes. For instance, Figure 5.11 displays the time-varying combustion dataset classified by the five relationships as in Figure 5.10 (b). Figure 5.11(a) is a traditional rendering of the tagged volume while Figure 5.11(b) is a rendering showing non-photo realistic effects. We also threshold the lowest scores to achieve

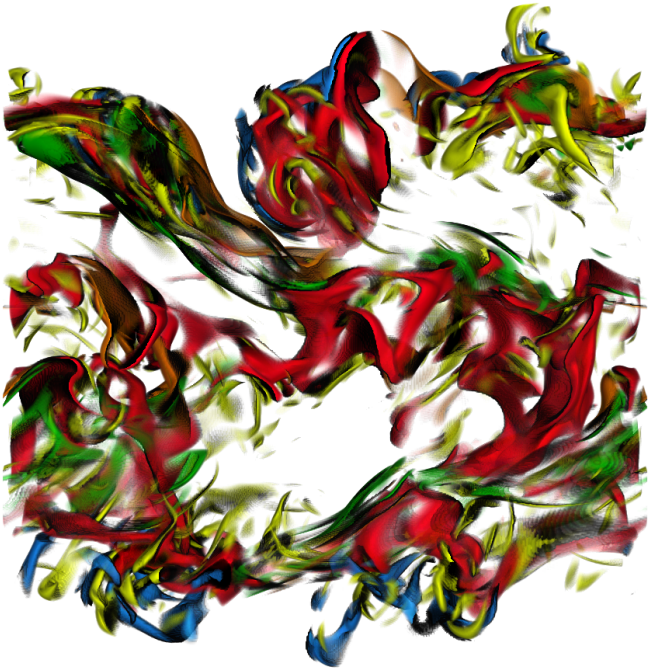


(a)



(b)

Figure 5.10: Images created from the Combustion dataset, timesteps: 116-117.



(a)

(b)

Figure 5.11: Sample 3D renderings.

visual clarity. We now detail our 3D renderer.

To interact with relationship volumes, we have built a GPU-based tagged volume renderer that uses both tag and score to determine a sample's color and opacity. Users can set the color and opacity for each tag interactively by manipulating a 1-D transfer function indexed by tag id. A second transfer function allows the user to modulate opacity by the sample's score. Typically, tagged volumes are shaded using the gradient from the presegmented volume. In our method of multivariate segmentation, however, no single gradient exists. Instead, we use the score volume. Since scores are spatially coherent and exhibit few high frequencies, this volume is ideal for gradient calculation.

We have also found it useful to increase the contrast between segment boundaries using NPR techiques. For segment-segment boundaries, we compute each voxel's distance to the closest (non-air) segment in a preprocessing step. For a thin boundary, only a fixed-size neighborhood of 5^3 voxels surrounding each voxel needs to be scanned. On the GPU, then, a sample's color is modulated to black if its distance is less than a threshold. Segment-air boundaries are also highlighted using silhouettes.

5.4 Discussion

We create images that represent multiple multivariate relationships. Since such relationships may be specified in any number of ways, there is likely overlap and/or simplification in the output. Images created from several relationships should be used only as a simple generalization of those relationships, i.e. a place to start. For more in-depth and specific verification of or discovery of relationships, it is recommended fewer relationships be rendered. Although we provide composite renderings, in the event of many relationships, these renderings may hinder the ability to distinguish among the relationships.

There is also a possibility that multiple relationships may equally well fit a region. In this case, the result may contain messy or random looking results. We find in practice, that these areas are not common, and may actually be points of interest in a rendering. Another issue with our approach is in the specification of relationships. If it is desired for relationships to be very finely specified, and with highly constrained or tight fitting of relationships, RS-files are not well suited for this task, i.e. the file itself may need to be generated via another program. Also, we select only

the best fitting relationships, we realize this may not always be the desired case. We believe that our approach is applicable here, although such a result may require some ingenuity on the part of the user.

Chapter 6

Image Space Classification of Parallel Coordinate Plots for Navigation and Rendering

The analysis of multivariate data is a difficult problem. Parallel coordinates is a proven focus+context tool that is useful in exactly this area. Parallel coordinates make for simplified detection of trends or similarities among chains of variables from a dataset. However, there are several shortcomings of parallel coordinates.

A visualization of parallel coordinates becomes increasingly difficult to analyze the greater of the number of variables rendered. In addition, parallel coordinates are very sensitive to the ordering of the variables. This is especially true when dealing with a large number variables, and in this case, it is necessary to aid the user in both the selection of the variables and in the ordering of those variables. Another problem with parallel coordinates is that to detect global trends in data, the full set of spatial points should be represented in each axis pair rendering. However, this commonly leads to very cluttered visualizations. Much research has gone into decluttering the image created between an arbitrary axis pair.

In this work, we study (i) the ordering/selection of axes for rendering and (ii) alleviating cluttered results while maintaining the full set of spatial points. The achievement of each of these goals is rooted in our use of image space measurements of parallel coordinates, i.e. metrics. Metrics have been widely used in parallel coordinates, primarily to measure the "goodness" of a

rendering before or after decluttering. Our metric designs take a different angle to instead focus on classifying parallel coordinates. Low vs. high metric values only distinguish differences, not which is "better".

We begin our design process by identifying a set of trends that are commonly expected in parallel coordinate renderings. We then design image space metrics to detect such trends by primarily characterizing white spaces. Using those metrics, we built an interactive system to navigate through parallel coordinate renderings to search for axis pairs with the desired visual trends.

There are also several other interesting uses of our metrics.

Metric based rendering: In the case there is a priori knowledge of a representative axis pair with desired visual properties, we can consider that pair to be a point in high-dimensional metric space, and simply create a rendering based on nearby points in that space.

Rendering via partitioning: Also, since each metric we provide classifies the set of axis pairs in one of two ways, we can use a binary search (i.e. recursive binary partitioning) to narrow down search space to find that representative axis pair for axis selection in rendering.

Rendering via directed decluttering: Lastly, a user may be interested in whether a certain type of relationship (i.e. a target pattern) exists between a specific pair of variables, but the parallel coordinate plot of the pair is too cluttered. In this case, we introduce a directed decluttering scheme that "numerically optimizes" the rendering such that its image space metrics approach the metrics of the targeted pattern.

In the remainder of this chapter, we first introduce our metrics for characterizing trends and demonstrate their direct use in finding parallel coordinate renderings in Section 6.1. Section 6.2 introduces the selection system framework and how the metrics can be used to detect trends. Then, in Section 6.3 we detail the decluttering approach used to identify desired visual properties in target parallel coordinate plots. Finally, our results and conclusions are provided in Sections 6.4 and 6.5 respectively.

The primary dataset used in this chapter is a 256x128 land/air climate simulation dataset with 63 variables. There are therefore, 1953 different axis pairs. For our renderings, we have removed any of the data items containing a single missing value, leaving approximately 8K records. Also, we have kept latitude and longitude as variables, in that their constant underlying structures possibly provide verification of results.

6.1 Image Space Metrics and Considerations

In this section, let us start with developing quantitative metrics to capture distinctive image space patterns between individual pairs of variables. The proposed image-space metrics measure specific visual properties that are important to users when attempting to comprehend a visualization. This capability facilitates the user's first need of quickly previewing the most unique trends in a new dataset containing thousands of variables.

Most of the proposed image-space metrics in this chapter rely on studying the white spaces (i.e. empty spaces) left in a parallel coordinate rendering. We first need to quantify the amount of remaining empty spaces (Section 6.1.2). Second, we would like to know how the white spaces are distributed. We have developed metrics that directly measure whether the empty spaces are scattered disjointly in the parallel coordinate rendering (Section 6.1.6), or they are compacted in a few larger contiguous regions (Section 6.1.5). We also propose complementary techniques to indirectly make such measurements (Sections 6.1.4 and 6.1.3). In Section 6.1.7, we discuss properties of image space metrics and display some results of using metrics for final renderings in Section 6.1.8. In the following, we begin by describing the common starting point of all our metric computations.

6.1.1 Using the Framebuffer

The problem of extremely cluttered images in parallel coordinates to a large degree is due to the screen sizes we are limited to when rendering. While it is possible to alleviate this constraint by increasing the resolution of the framebuffer, the cost to render the plot and compute image space metrics increase accordingly. We have used a 400x800 framebuffer for each pair of axes, we had found it sufficient in all of our experiments.

A primary goal of using metrics computed from image space is to convey some measure of the actual information perceivable by the user. It should be noted that while these metrics may be used to select variable pairs with the least amount of visual clutter, for some users, a highly compact and cluttered parallel coordinate rendering may reveal the most important trend (i.e. a pair that has no linear relationship). In the following sections we describe how to compute the proposed metrics along with explanatory images using a white background and black line renderings.

6.1.2 Open Space Metric

The simplest such metric is the open space metric ($open\_metric$). This, as it sounds, is the percentage of space between two axes not occupied by lines of the parallel coordinates system. By itself, the $open\_metric$ is useful as a litmus test for whether or not trends may be detected. However, in the simple example of Figure 6.1 , the shortcomings of this metric quickly become apparent. In these cases, the open space is near 50%, even though each is fundamentally different from the other. Interestingly, if no lines overlap, existing decluttering metrics also fail to differentiate between the left and middle images Figure 6.1.

6.1.3 Space Fill Difference

We now present our metric that captures how spread throughout the space lines appear in a parallel coordinate plot with the space fill difference metric ($diff\_metric$). The first step in the calculation of $diff\_metric$ is counting the number of all filled pixels that have an empty pixel above it. This count is then divided by the open space between the axes. Therefore, this metric is the percentage of open space left after increasing the width of each line by a pixel's width. In the first two images of Figure 6.1, $space\_fill$ is 100% and 20% respectively. This metric, however would not differentiate between the middle image in Figure 6.1 and a pair of axes with a single line. For rendered data, this metric indeed indicates whether or not the data is spread throughout the space.

6.1.4 Maximum Rise Metric

Another image space metric, which we refer to as $rise\_metric$, is the maximum number of consecutive vertical pixels that are filled in by lines of the parallel coordinate rendering. This is capable of discerning between the first two example plots in Figure 6.1 where the first image has a maximum rise of 1 and the middle image has a maximum rise of 5. Clearly however, this metric would also yield the same result between axis pairs such as the middle and right parallel coordinate plots in Figure 6.1, both of which share the same value with an axis pair containing only a single line.

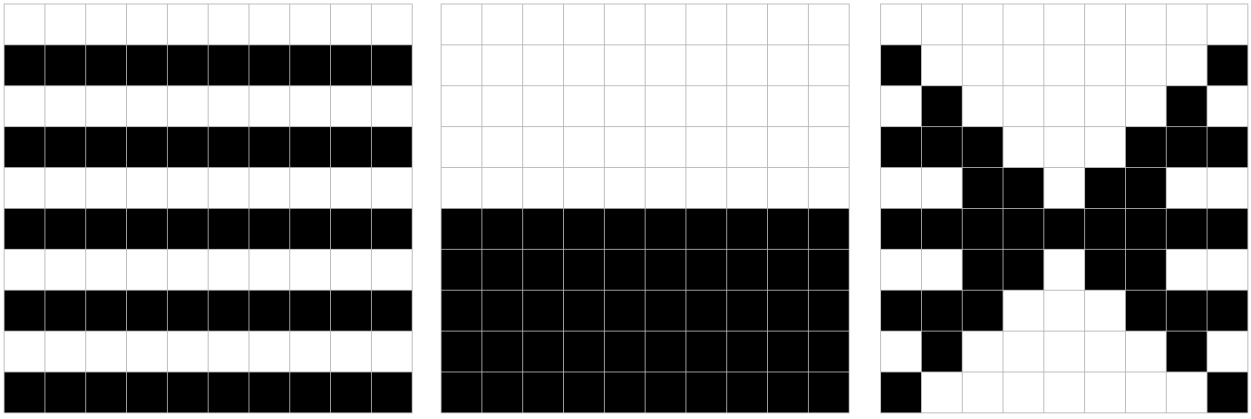


Figure 6.1: Illustration of patterns which can be disambiguated with the different image-space metrics proposed.

6.1.5 Maximum White Rise Metric

This metric is the logical counterpart of maximum rise in that it counts the maximum contiguous vertical span of empty regions rather than filled regions. A small value, relative to the total vertical dimension of the parallel coordinate rendering, would indicate that the data is spread throughout but is best used in combination with other metrics. For example, maximum $white\_rise$ is 5 for both of the middle and right images in Figure 6.1 and thus cannot disambiguate between the two. Large values of maximum $white\_rise$ and maximum $rise\_metric$ together strongly indicate the existence of clustering effects with at least one of the two variables.

6.1.6 White Span Difference

Lastly, we present a metric that is calculated by finding the maximum and minimum number of disjoint white spans for all vertical column of pixels, then taking the difference between the two. For the middle and right parallel coordinate plots in Figure 6.1, the maximum number of white spans is 1 and 5 respectively. The minimum number of white spans is 1 and 2 respectively, leading to a $diff\_rise$ value of 0 and 3. Due to similarities in computation, we compute this metric in the same pass that we compute max_rise and maximum $white\_rise$. We developed this metric to highlight the existence of a fanning effect in a parallel coordinate plot, i.e. a large area of spread lines that converge a single point somewhere in the axis pair space.

In Figure 6.2 we show the axis pair for each metric out of the 1953 possible axis pairs that correspond to the maximum and minimum metric value.

6.1.7 Properties of Image Space Metrics

We find that in practice, metrics calculated in image space have the property that the extreme values, both maximum and minimum, indicate visual properties of the space. For this reason, these metrics may be independently combined to further narrow a search through a parallel coordinates system. In the most general sense, one could consider nearly any image space metric to split a space into two separate classifications. Therefore, the combination of n image space metrics allows for 2^n separate classifications of a parallel coordinate space. In this way, axis ordering can be performed via very specific tendencies of preferred user views. Just as various combinations of "high" and "low" values for specific metrics "mean" different things according to the previous

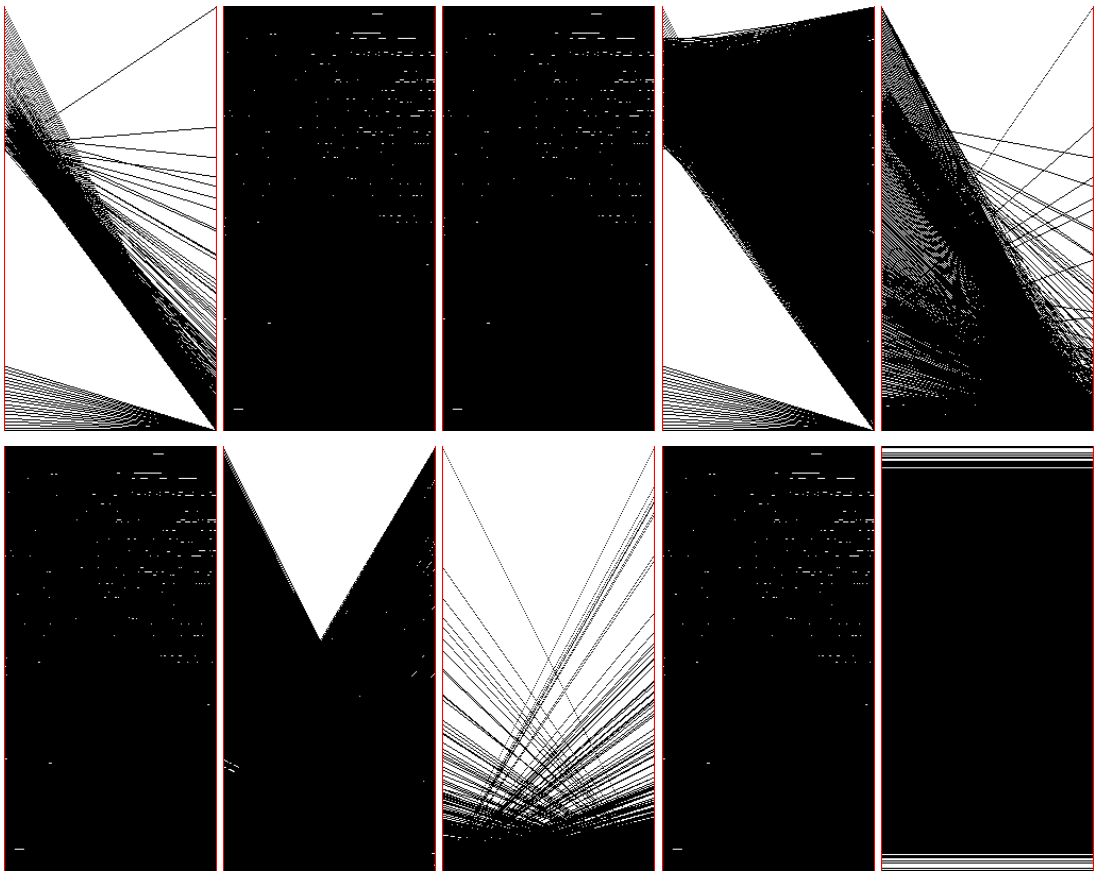


Figure 6.2: Max (top row) and min (bottom row) for each metric, from left to right: $open\_metric$, $diff\_metric$, max_rise, $white\_rise$, and $diff\_rise$ respectively.

sections, we allow the user to define which metrics they wish to be high or low and create a layout in which the variables with the max or min values for those corresponding metrics are used to arrive at a very specific set of visual properties.

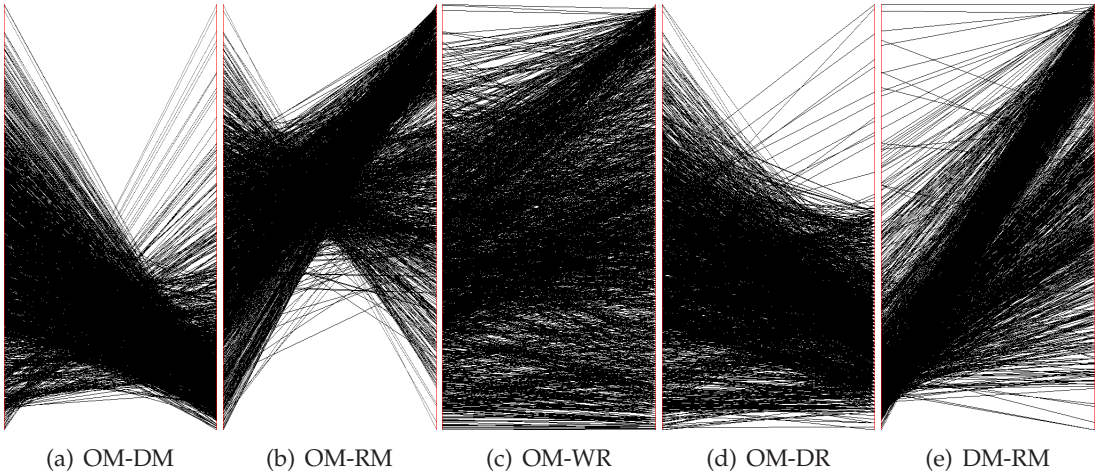
Although 2^n may in fact accurately represent the different number of classifications provided by a n metrics, there are noteworthy caveats. A metric's binary classification of a system is user dependent, and the range of each separate classification is unpredictable, if not empty. It may seem that an optimization routine, or at a finer level a decluttering process may exist on a function that maps a pair of axes or a set of lines into the n-dimensional space, but this is not the case. The n-dimensional space created by metrics does not contain independent dimensions, and it is also not possible to create a simplex in that space, i.e. there is no direction of an optimum.

Figure 6.3 shows parallel coordinate plots of the 10 different possible pairings of metrics to illustrate the unpredictability of the relationship between any pair of metrics. Every data record in this rendering corresponds to one of the 1953 axis pairs.

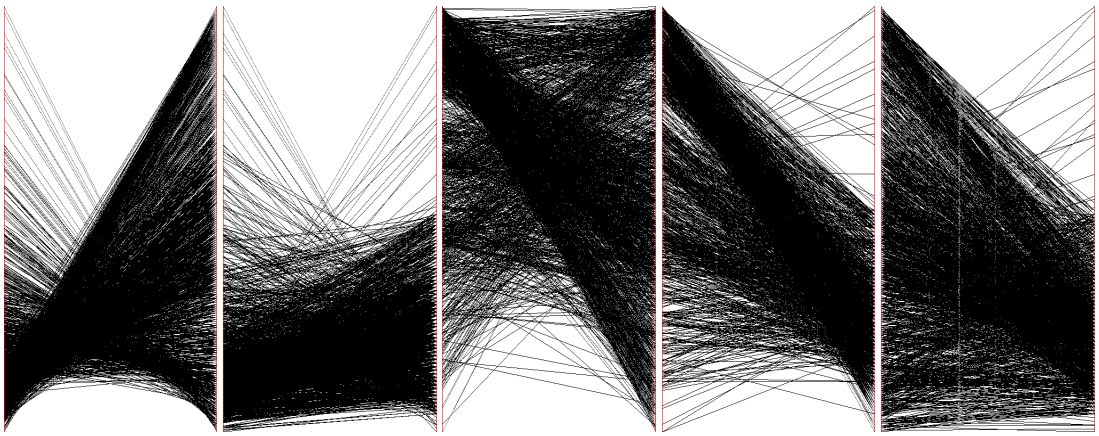
6.1.8 Using Metric Space for Rendering

In this section we explore the possibility of creating parallel coordinate renderings based solely on the metric space around a parallel coordinate plot of interest. Since each axis pair corresponds to five separate metrics, we treat each axis pair as a point in five-dimensional metric space. After normalizing each metric so that they are in the same range, we find the four axis pairs nearest in euclidean distance to the original in metric space to create a rendering. Note, in selecting five axis pairs this way, there is no guarantee the axes will match for a rendering. For instance, selecting two axis pairs for rendering, variables 1 and 2 and variables 3 and 4, a rendering of three axis pairs must be provided to account for the space between our two chosen pairs. Therefore, as in Figure 6.4, although we have only chosen a total of five pairs, the rendering is of ten axes. Remaining results typically alternate between selected pairs and resulting between pairs. In some cases, there are less axes. For these, two pairs of selected axis share the same axis eliminating the need for another between.

Figure 6.4 is the result of choosing a representative axis pair (the left-most in each rendering), and finding the next four closest in metric space and rendering these pairs along with the spaces between. For this example we have purposely chosen pairs with very distinct (although



(a) OM-DM (b) OM-RM (c) OM-WR (d) OM-DR (e) DM-RM



(f) DM-WR (g) DM-DR (h) RM-WR (i) RM-DR (j) WR-DR

Figure 6.3: Parallel coordinate renderings of pairs of metric values. Axis pairs are labeled by abbreviations of metric names: $open\_metric$(OM), $diff\_metric$(DM), $rise\_metric$(RM), $white\_rise$(WR), and $diff\_rise$(DR).

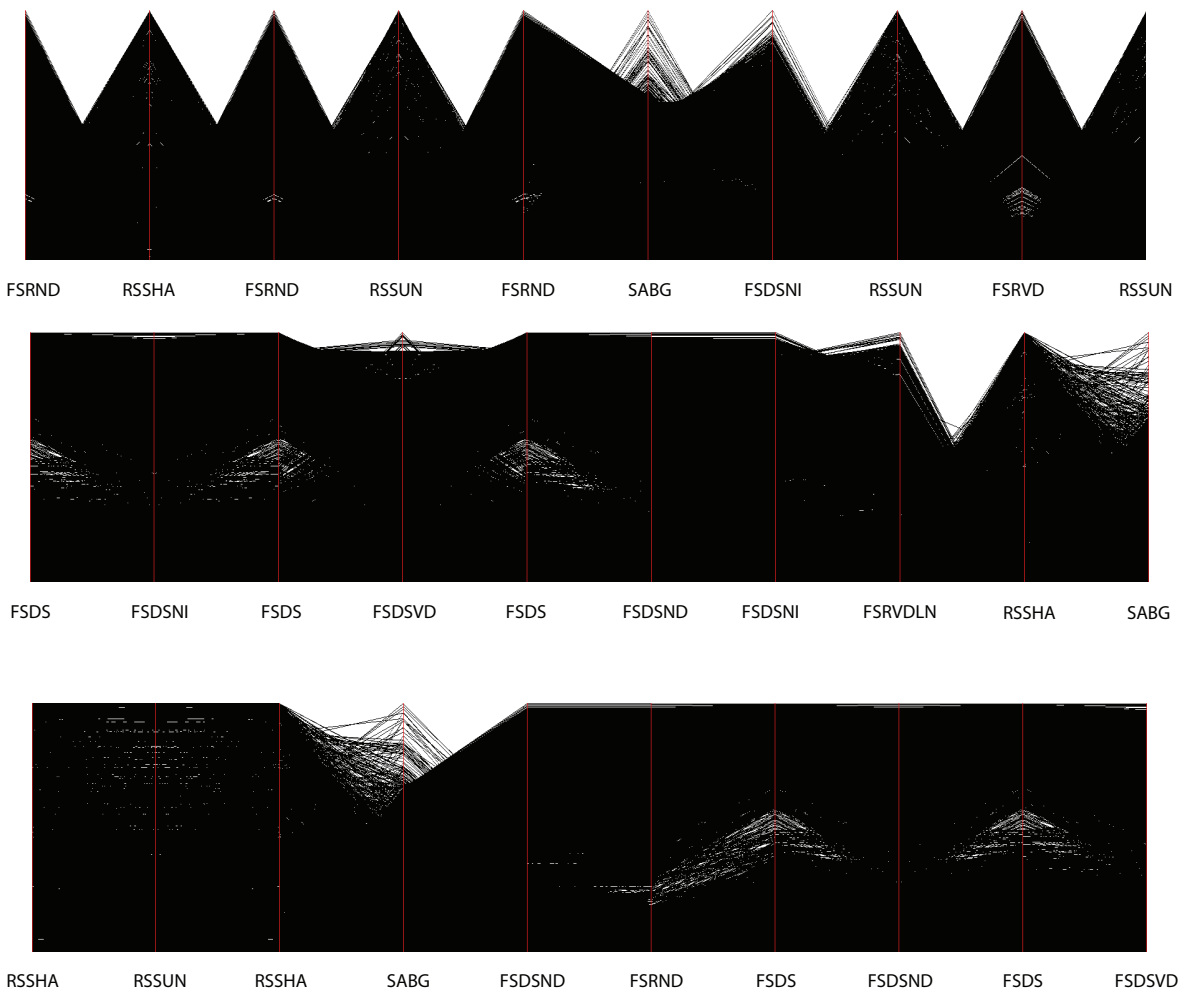


Figure 6.4: Parallel coordinate renderings in each are row produced from the left-most axis pair. This pair is treated as a point in metric space, then the remaining axis pairs are those closest in that space.

not interesting) visual properties. The resulting renderings contain sets of pairs with similar visual properties to the representative pair, even those pairs between the selections, a phenomenon we will investigate further in Section 6.4.

6.2 Navigating via Binary Partitioning

In this section we describe a system to quickly and interactively let the user find many different parallel coordinate renderings that are of interest, especially in the event that there is no a priori user preference. The system is designed to be both general and interactive for the selection of parallel coordinate axes.The system leverages a set of metrics that are developed to classify the rendering of an axis pair. The images displayed by the system are a representation of all images in terms of the way they are classified based on a specific metric. In Section 6.2.1 we discuss the system considerations for generating metrics and ensuring system interactivity. In Section 6.2.2 we show our system on startup and discuss the various displays/controls.

6.2.1 Preprocessing

To ensure the interactivity of our system, much of the necessary calculations are performed during preprocessing. The metrics we use for classifying parallel coordinates are those calculated in image-space. For this reason, we calculate all possible frame buffers of axis pairs and calculate all metrics based on those images. To avoid rendering costs during system interaction, we store those images on disk, and reference them directly from the system. The dataset used throughout this chapter contains 63 variables, therefore 1953 possible axis pairs and images. These images, when stored as GIFs, require less than 7MB of disk space.

A user may supply any set of images and metrics to our system. We read a set of metric names from a file, then look for separate files containing all calculated metric values when appropriate. In this way, any metric calculated between pairs of axes is appropriate to be specified for our system. In addition to generating possible metrics, calculating images as a preprocessing step has another advantage, any parallel coordinate rendering software may be used.

6.2.2 Interactive Exploration

Figure 6.5 shows our system upon normal startup, we will now describe its interface.

The major display is that of 3 renderings of axis pairs. These represent the entire set of axis pairs, when sorted by a specific metric. The first metric used for sorting the images, is whichever is specified first in the file that lists the metric names (see Section 6.2.1). The first and third images are the two axis pairs corresponding to the highest and lowest value of the metric used for sorting, respectively. The rest of the images are represented with the horizontal slide bar at the top of the system, with the middle image corresponding to the slider's position.

In general, there are two problems with classifying an axis pair rendering. Firstly, a metric doesn't evenly cleave the set of images, for instance, in the case of the open space metric, ninety percent of all images may be too cluttered. The second problem is that what exactly the metrics classify is not exclusive to each metric, i.e. low values of $diff\_metric$ and high values of $open\_metric$ may contain similar images.

For these reasons, the purpose of the system then, is to interactively classify images of axis pair renderings leveraging a user's interpretation of images to address the two problems of classification. For a given metric the user selects the middle image by using the slide bar, the idea being that all images between the first and middle are classified one way, while images between middle and last are the other. The right and left text boxes at the top of the system inform the user as to how many images are in each of these ranges.

In Figure 6.5, the first metric is $open\_metric$ which is seen selected in the drop down menu. This menu is used to select the metric for sorting. Once the user is satisfied with the metric chosen and a classification, the user may narrow the set of axis pairs by pressing $SelectLeft$ or $SelectRight$, i.e. choosing the range between first and middle image, or middle and right, respectively. The $RenderRight$ and $RenderLeft$ both output a list of sorted axis pairs so that a custom rendering may be created. For all results in this chapter, we select the top five axis pairs, and use the same renderer used for the axis pairs to create the renderings. The selection of five axes is arbitrary, and done for the purpose of displaying results.

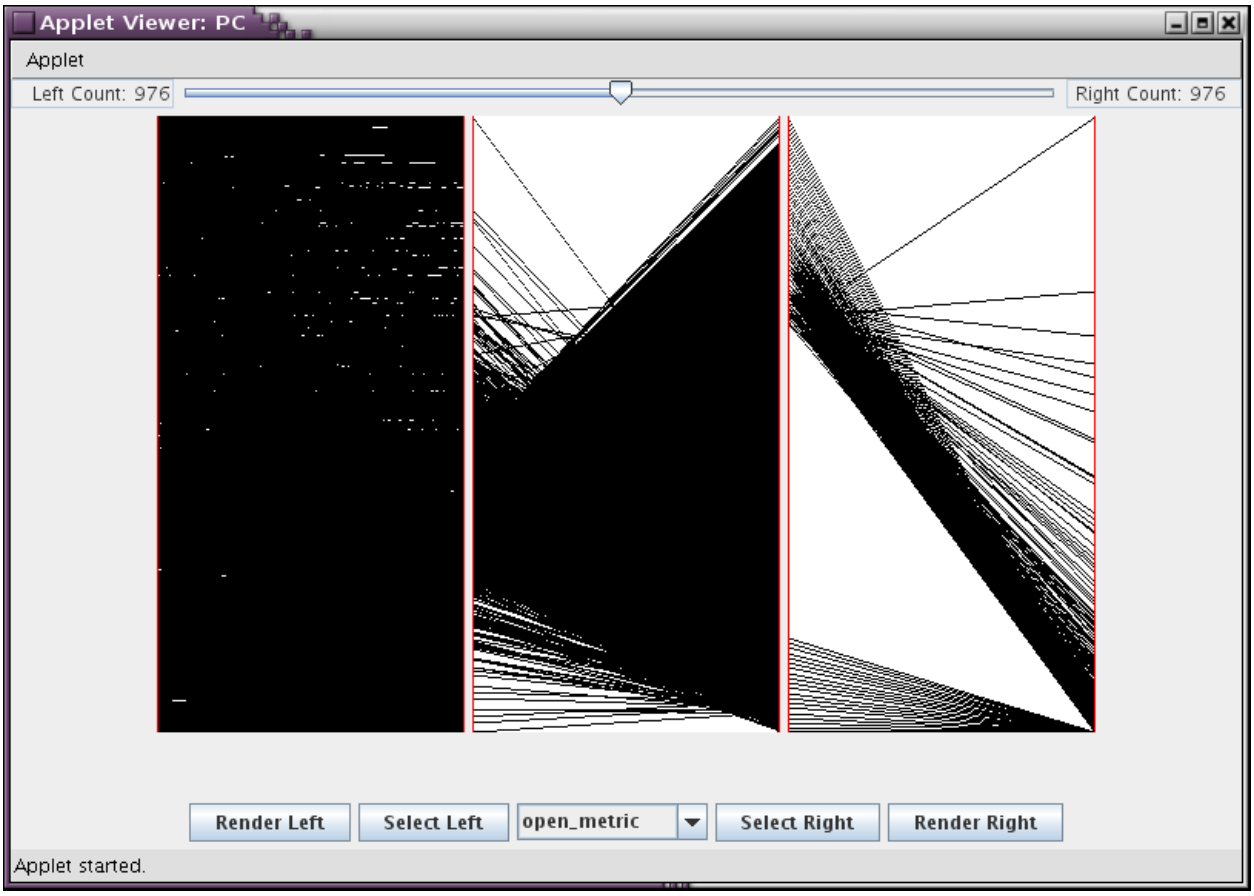


Figure 6.5: The system on startup.

6.3 Directed Decluttering

In this section we detail a decluttering approach that leverages Section 6.2's system in two ways. First, we use the system to select a target parallel coordinate plot that exhibits some set of desired visual properties. We then take a different parallel coordinate plot, typically a more cluttered one, and declutter it in a way that leaves a parallel coordinate plot with similar visual properties to the target rendering. Specifically, we are attempting to declutter the same axis pairs that were used for axis selection in Figure 6.4.

6.3.1 The Approach

For such a routine to work, there must be a measure of distance between two parallel coordinate plots in metric space. As in Section 6.1 the metrics are normalized and this distance is simply euclidean distance. For the remainder of this section, we refer to the axis pair containing the desired visual properties (the target) as T, and the second pair as S, the plot to declutter.

Figure 6.6 displays the general psuedocode for our decluttering scheme.

There are also a few additions to this code, i.e. changes we have found that greatly improve the performance of the decluttering routine. The combination of multiple metrics can make for a confusing classification. For this reason, and also since it is sometimes clear there is a large disparity among S and T in terms of a single metric, we allow the user to select a single metric and then run the routine in Figure 6.6 for only that metric. For the results in this chapter, we have consistently left this as $open\_metric$. We then perform the algorithm in Figure 6.6 as is. Since the order of removal may play a role in the performance of the decluttering approach, as a final step, we go through the set of all removed lines and re-add any that makes the distance between S and T smaller. Although this step was added to avoid possible sensitivity to the order of line removal, we find that in practice, this overall approach produces visually indistinguishable results regardless of where the selection is seeded to start. However, the addition of this step does improve the performance of the decluttering routine, in terms of the quality of resulting renderings.

We show some sample results of running this decluttering procedure in Figures 6.7 and 6.8. In each of these rows the first two images from left to right correspond to our chosen S and T respectively, and third is the decluttered S. For the first two rows, we selected what appeared to be very cluttered axis pairs that could have similar visual properties as the target. For the last two,

```
for i := 0 to number of lines in S
  select random line, L, from S
  S := S - L
  calc_metric(S)
  if S closer to T
     continue
  else
     S := S + L
  endif
endfor
```

Figure 6.6: Decluttering psuedocode.

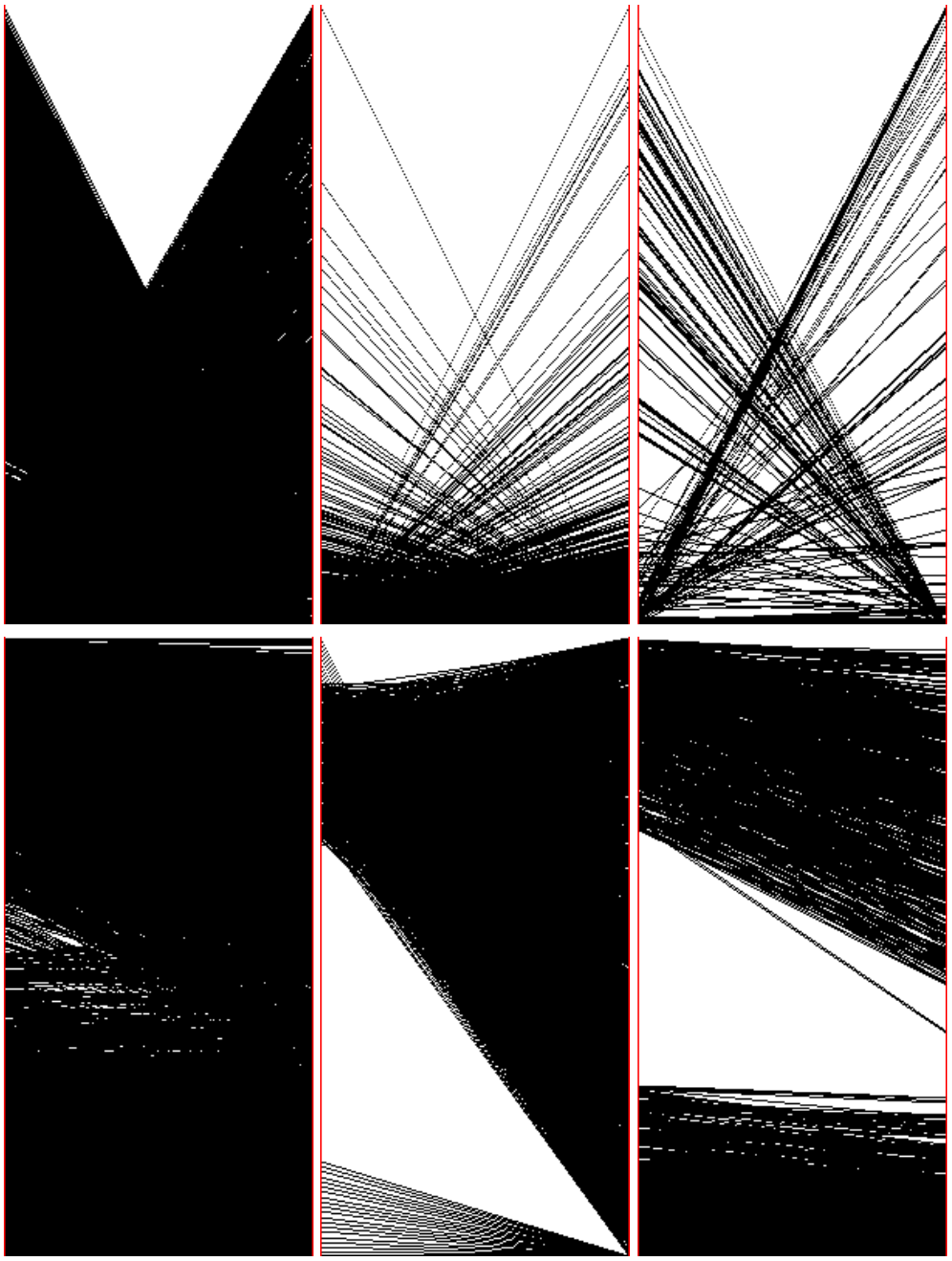


Figure 6.7: Examples of the decluttering scheme. Each row represents a separate run. The first image in each row is S, the second T, and the third is the decluttered S.

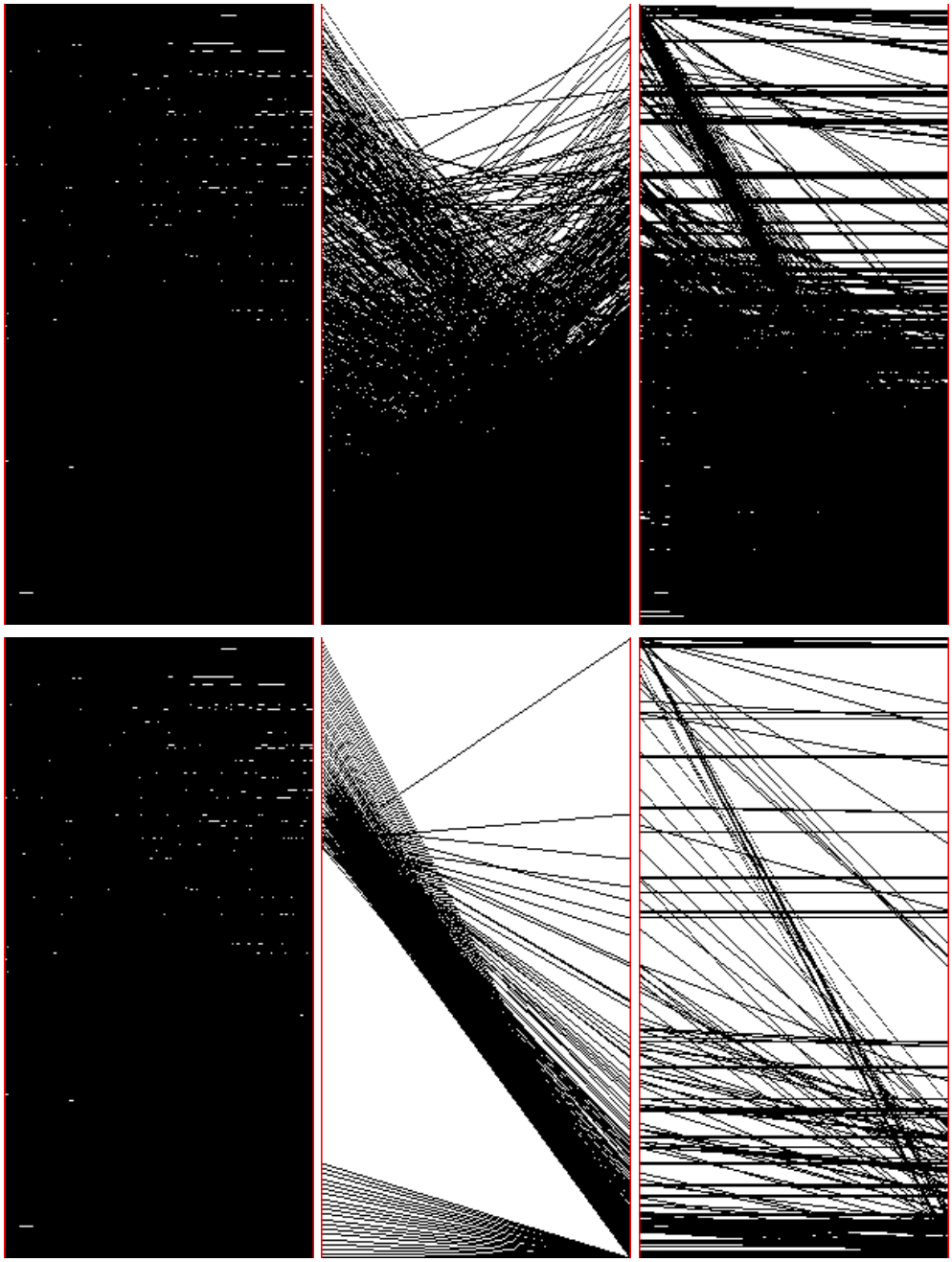


Figure 6.8: Two more examples of the decluttering scheme.

we decluttered the axis pair with the minimum white space among all axis pairs.

6.3.2 Performance and Analysis

This decluttering routine makes two full passes through the entire set of lines in a parallel coordinate plot. Each pass requires a rasterization of the framebuffer and a recalculation of the metrics. Our rasterization code is linear in terms of number of lines and framebuffer size, and the metric code is linear in only framebuffer size. We decreased the framebuffer size for these calculations to 200x400 and on a single processor workstation, the rasterization code completes in $.05$ secs and the metric calculations are instant. Therefore, as worst cast scenario, on this 8K data record dataset, this routine would take approximately 13 minutes. We find in practice, each iteration is performed on a smaller number of records and two full passes is unlikely, therefore runtimes are closer to around 5 minutes.

As we have demonstrated in Figures 6.7 and 6.8, our decluttering approach does indeed render a parallel coordinate plot with specific visual properties, even if those properties are not readily apparent in S before the decluttering. In particular, the approach illustrates different appearances of parallel coordinate plots that exhibit similar visual properties. The accuracy of the results of this approach in terms of visual properties are proportional to the quality of the set of metrics' measurement of such visual properties. We feel that in conjunction with the system provided, the decluttering routine is not only a way to search for specific trends, but also an excellent measurement of the performance of generated metrics. This gives a user a more complete picture of what a metric actually classifies.

In the rows of Figure 6.8, the decluttered S and T are near each other in metric space, but there are certainly visual properties not shared between the two. In this case, there are only very few lines in S with a high positive slope, so an exact visual match would not be possible. However, the output does steer the user in the direction of what additional properties must be accurately described by metrics to improve the visual similarity of two coordinate plots, rather than producing two with similar visual properties. Visually, the decluttering appears to perform well on the selected axis pairs.

We now verify, with an empirical study, that the approach does not change the underlying structure of the parallel coordinate plot during the process of decluttering. The results of this

study are shown in Figure 6.9.

We have created a parallel coordinate plot with 500 records, shown in Figure 6.9(a). We then add several series of random numbers to this plot, ranging from 40 to 4000 lines. The percentages under the remaining images on the top row are how many of the lines are from the original dataset before adding random lines. The bottom row is after decluttering, and the percentages under the images correspond again to the number of lines from the original plot. Since adding random lines does not change the underlying structure of the plot, a successful decluttering routine would maintain similar or slightly higher percentages in the bottom row as in the top. Note, it is visually clear that our metrics do not do a very good job modeling the structure on the bottom half of the original plot, not being able to distinguish it from any other space of cluttered lines, and therefore the good performance statistically in the last columns are due to the encapsulation of the upper structure.

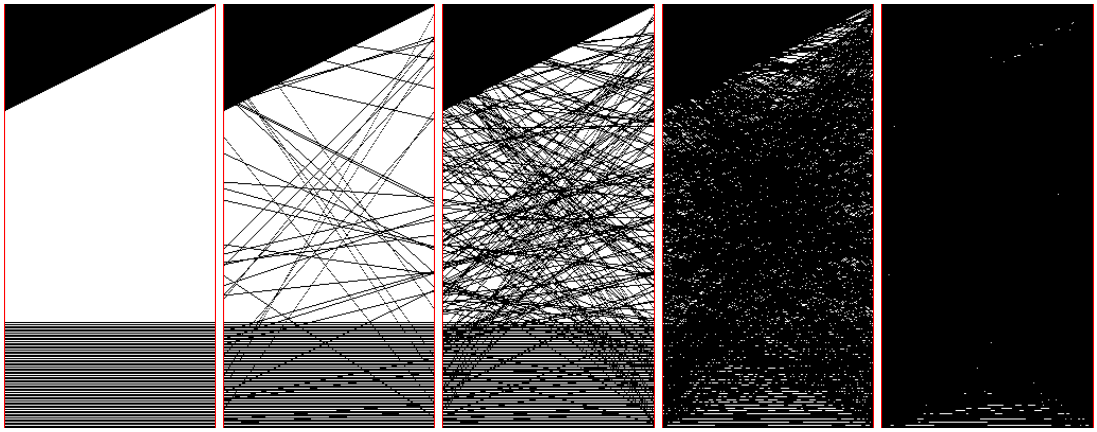
6.4 Results and Analysis

In this section, we present example outputs from our system presented in Section 6.2; we consider these images (Figures 6.10-6.13) to be representative of certain sets of visual properties. We then show sample renderings of those results in Section 6.4.2. In Section 6.4.3 we show renderings created from the same axis pairs as those in Section 6.1, except based on the desirable underlying structure of those pairs. Lastly, we end with a discussion of the pairs between those selected for rendering in Section 6.4.4.

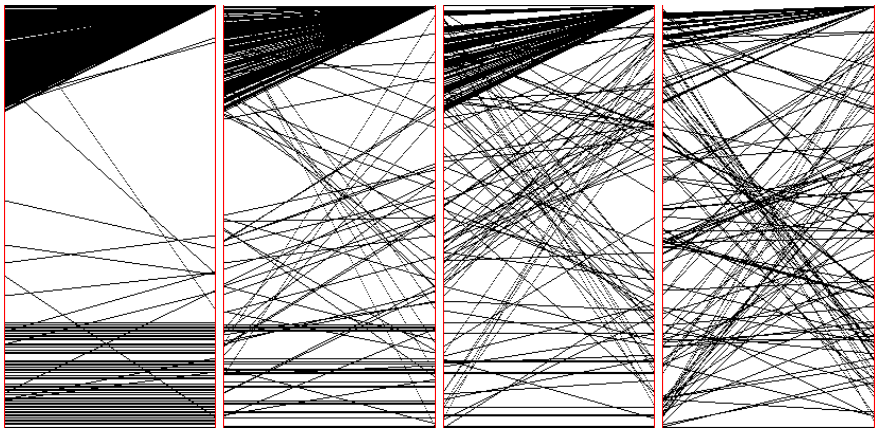
6.4.1 Navigating Parallel Coordinate Plots

We now display representative images of arbitrarily chosen visual properties, as output by our interactive system. We use Figure 6.5 as a starting reference for the images in this section. Figure 6.10 is the result of pressing $SelectRight$ and resorting with the $diff\_metric$. The first and last image in Figure 6.10 correspond to least/most spread out parallel coordinate plots respectively.

However, since a selection was made based on $open\_metric$ previously, this is within the context of the top 50% of axis pairs with respect to the amount of open space in their parallel coordinate plots, i.e. the set of axis pairs where different trends are more readily apparent.



(a) (b) 92.59% (c) 71.43% (d) 33.33% (e) 11.11%



(g) 91.52% (h) 71.52% (i) 50.00% (j) 37.67%

Figure 6.9: Decluttering structure analysis. (a) Synthetic dataset created with two structures, each consisting of 250 lines. (b)-(e) The appearance of the dataset after adding 40, 200, 1000, and 4000 random lines, respectively. (g)-(j) The resulting plots after the decluttering routine.

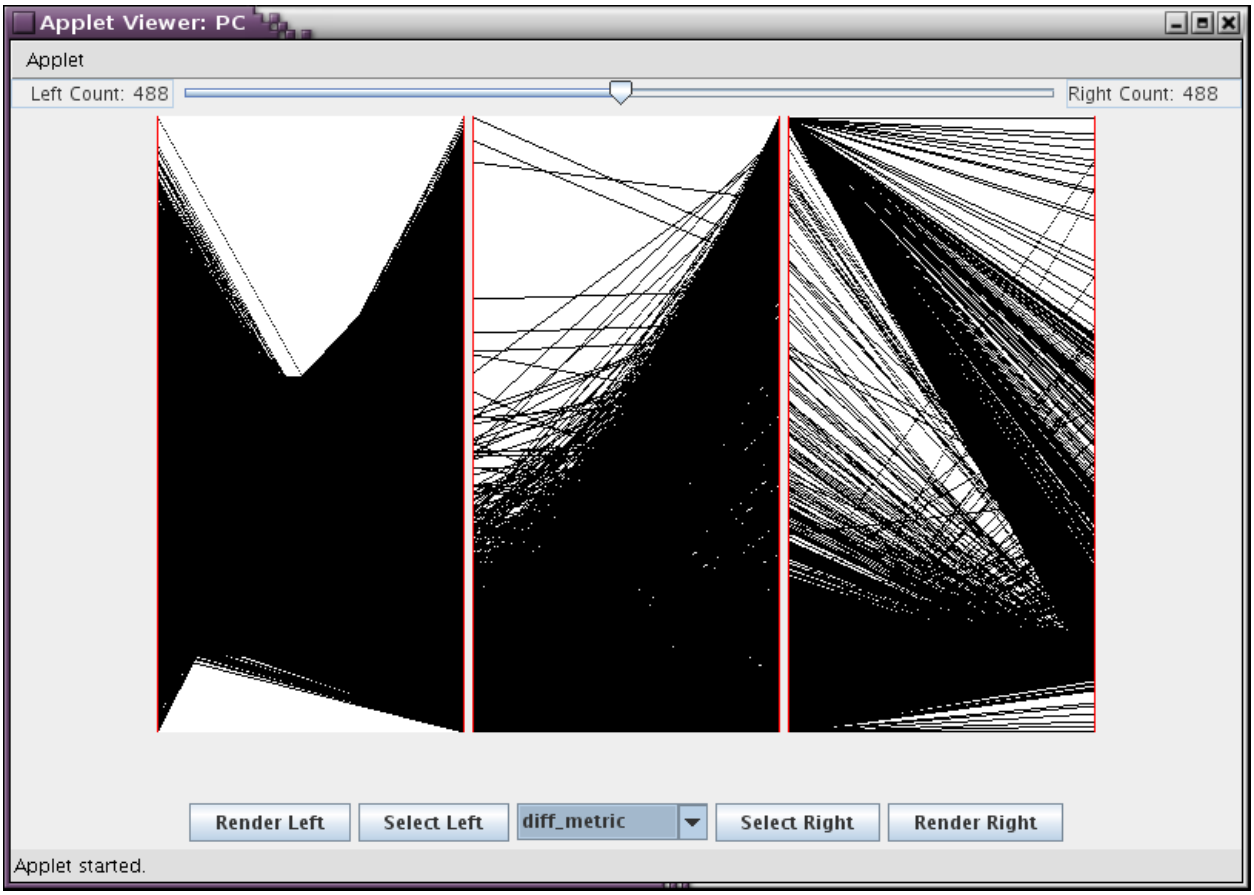


Figure 6.10: After performing a selection, only the axis pairs with the selected metric values between the middle image and the left/right (depending on selection) remain. After selecting the axis pairs representing the top 50% with most white space, these are sorted by $diff\_metric$.

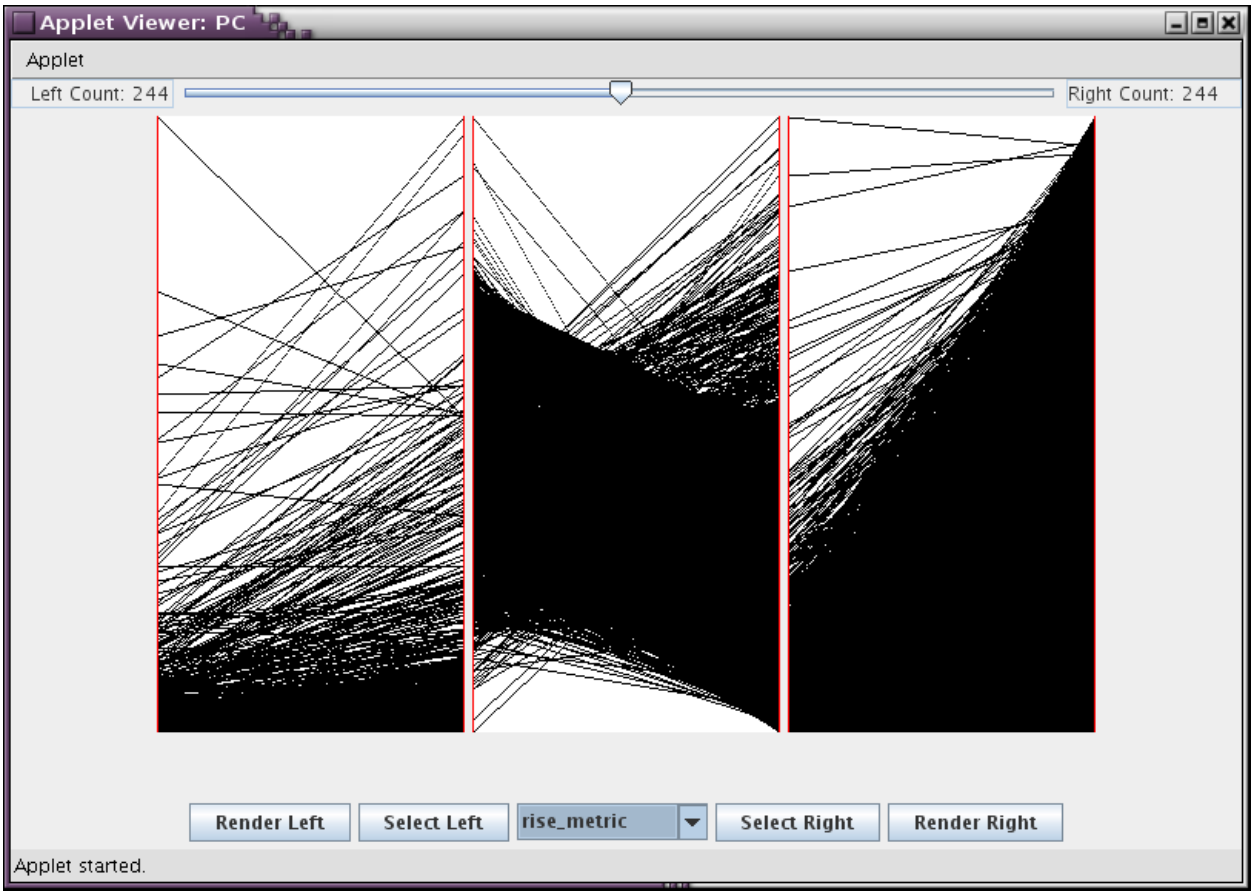


Figure 6.11: Remaining axis pairs sorted by $rise\_metric$.

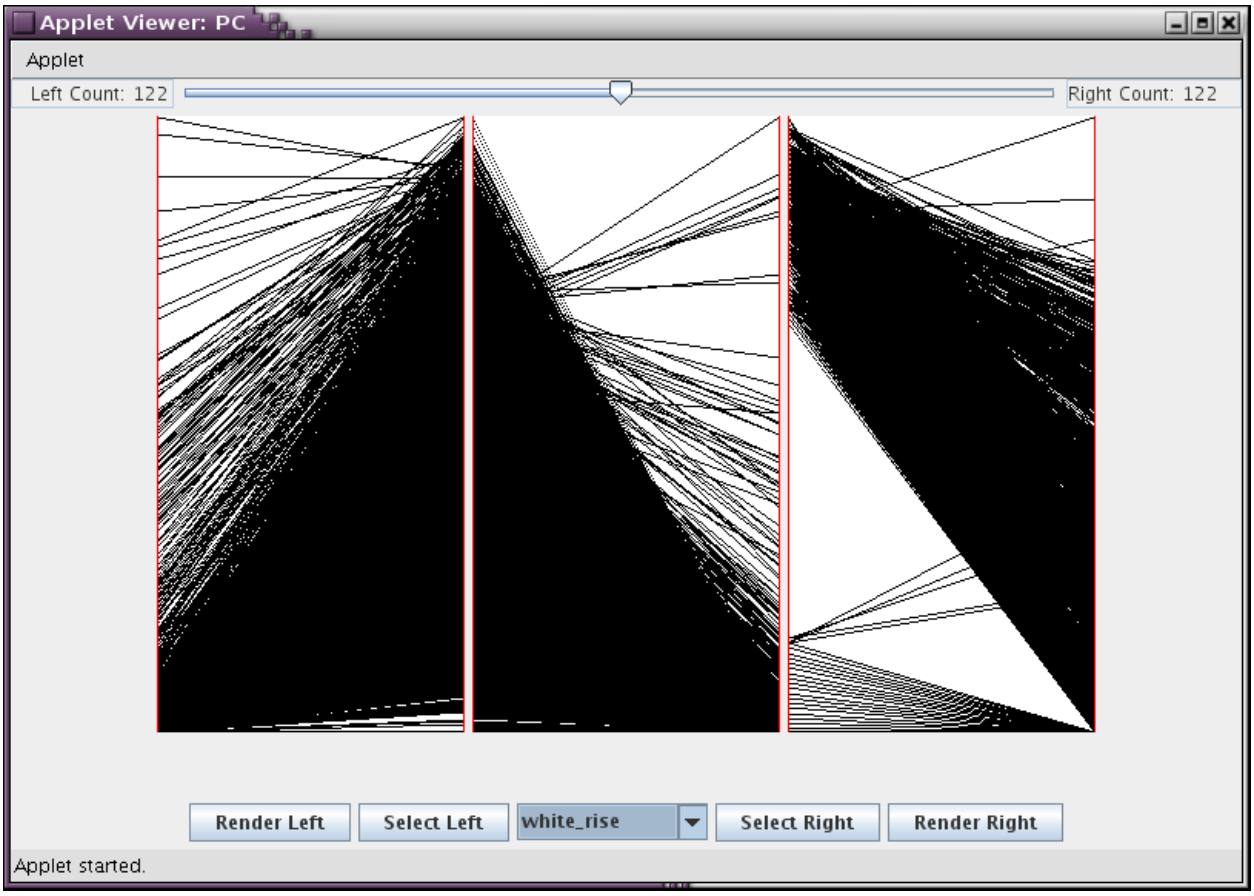


Figure 6.12: Remaining axis pairs sorted by $white\_rise$.

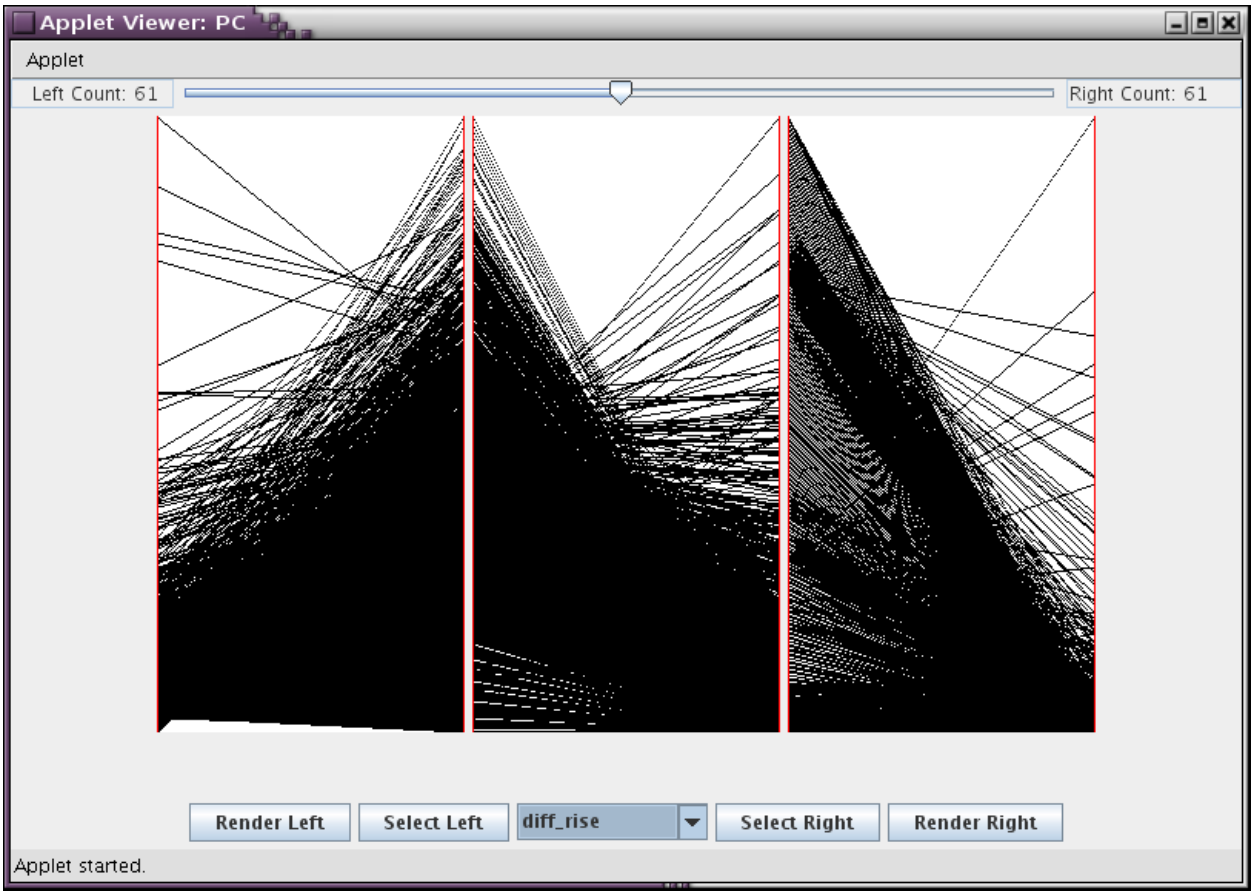


Figure 6.13: Remaining axis pairs sorted by $diff\_rise$.

For the remaining images, we chain together metric navigation by performing another $SelectRight$ before every new sort. So, for Figure 6.11 we are sorting the top 50% of axis pairs with lines spread throughout the space still within in the context of the 50% of axis pairs with the most white space, etc. Next we $SelectRight$ and sort by $white\_rise$, then one more $SelectRight$ and the last sort: $diff\_rise$. Narrowing the number of images hones in on some set of specific visual properties and creates a more complicated classification of the remaining images.

After the final selection/sorting there are only approximately 120 axis pairs remaining from the original 1953 axis pairs. Also, it should be noted that the axis pairs could have been further narrowed by moving the slider at any step, and the axis pairs could also be resorted and narrowed by any metric again after Figure 6.13.

6.4.2 Rendering from the Navigation

Figure 6.13 shows a representative image of the parallel axis pairs after they are sorted based on a very specific and complicated classification of all axis pairs. The top row of Figure 6.14 corresponds to selecting that pair along with the next four in that sorted list, and creating a rendering along with the pairs existing between them. Interestingly, all five selected pairs share the longitude variable. Since the last sort is based on the difference of maximum and minimum white rises, the top axis pairs are all those in which variables have similar values across the entire spatial region. This is the reason for the very similar visual properties across the rendering.

Since this classification is so complicated, we take a step back and look at a similar one, for the ease of interpreting the results. The last two rows of Figure 6.14 are renderings of the top and bottom five images from the sorted axis pairs created in Figure 6.10. The second row corresponds to the bottom 5, the space of the 50% of the axis pairs with the most white space but the least spread out through the framebuffer. The last row corresponds to the top 50

6.4.3 Rendering from Decluttering

In Section 6.1 we created renderings from selected axis pairs. In Section 6.3 we took those same pairs and decluttered them in an attempt to discover certain visual properties within them. In some cases more than others, those properties did exist in the selected pairs. In this section, rather than using the metrics calculated from those selected pairs, we take the metrics from the declut-

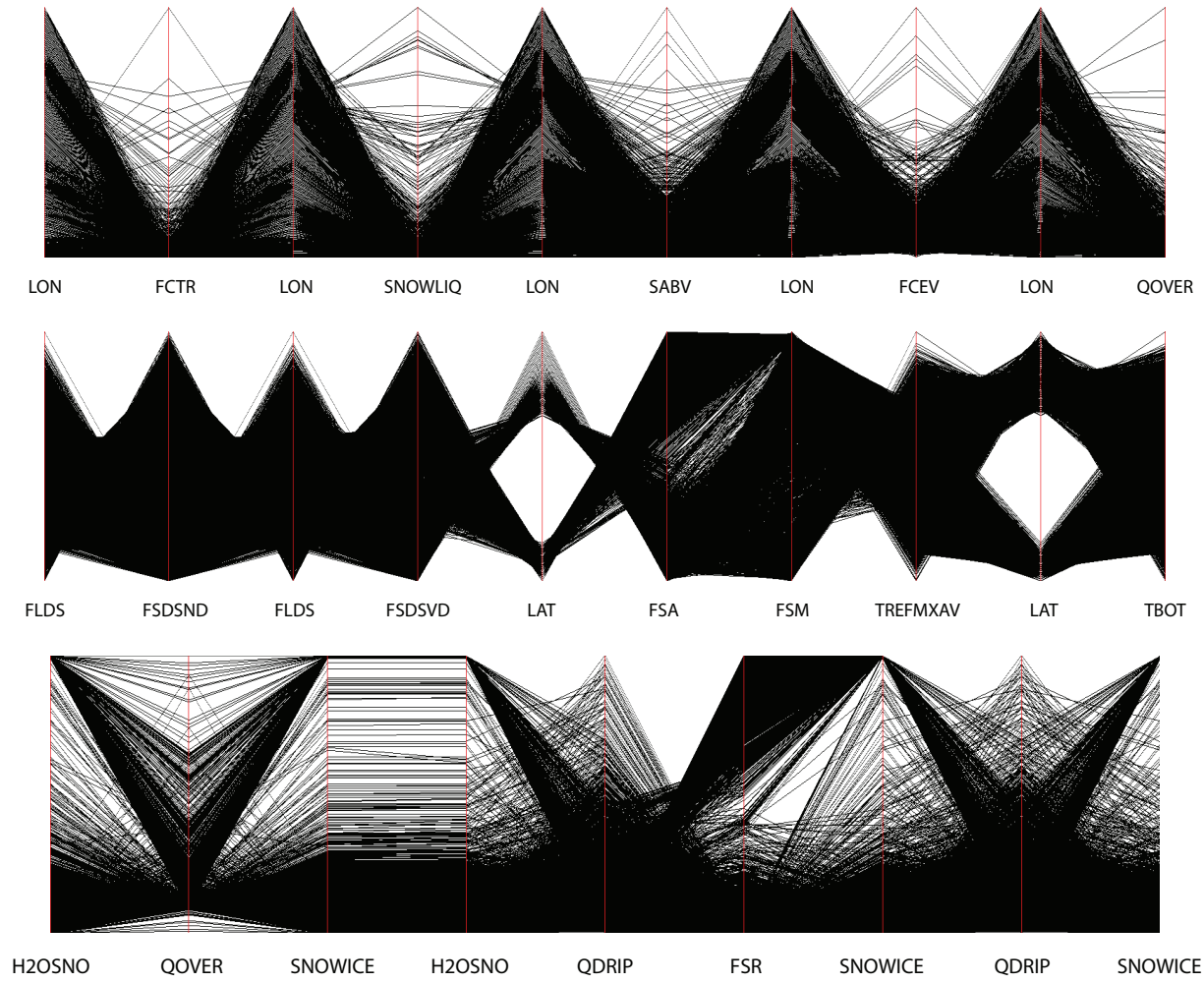


Figure 6.14: Final renderings from the system. The system was used to navigate to the left-most axis pairs in each row. These renderings are then created by selecting the next axis pairs in the sorted list created by the system.

tering results and use those as the elements of the metric space point to start the axis selection. For each of the rows in Figures 6.7 and 6.8 we have created such a rendering, these are the four rows in Figure 6.15.

Notice, the left-most image is still the original as in Section 6.1. Although these renderings show completely different trends than those in Figure 6.4, there is actually still no decluttering done in the final result. Therefore, this displays the way in which these trends propagate from the original axis pairs. The last two renderings are of nine axes, this is due to the fact that in those two cases, two of the five axis pairs actually shared an axis, and therefore there was no need to render a pair between them.

6.4.4 The Spaces Between Selections

In this section, we perform an empirical study of the spaces between our selected axis pairs. Specifically, in all of our final renderings, although we are only selecting five axis pairs, are renderings of up to ten axes. Therefore we are having to render axis pairs that are in no way guaranteed to be close to the representative axis pairs that the renderings were created from. This is the case whether renderings are created from metric space or from classification by metrics. Surprisingly, those axis pairs between have similar properties to the five axis pairs numerically selected. This suggests the logical property that axis pairs are closer in metric space to other pairs where variables are shared.

We calculate for each axis pair, the average distance, d_1, between it and every other axis pair, and then calculate the average distance, d_2, between it and only the other axis pairs with a shared variable. For this, we still use euclidean distance over normalized metrics. For our climate dataset, it turns out that for 94.32% of axis pairs, d_2 is smaller than $d1$. With such a high number, we compare with another dataset: a 12 variable, 500 record astronomy dataset. For that dataset, 96.97% of axis pairs have this property. Lastly, we calculate the average rate that d_2 is better than d_1, (d_1/d_2). For the two datasets, these average rates are 1.22 and 1.21.

6.5 Conclusion

In this chapter we reexamine potential uses for metrics in parallel coordinate renderings. We extend their traditional utility of decluttering parallel coordinate plots to quantitatively measuring

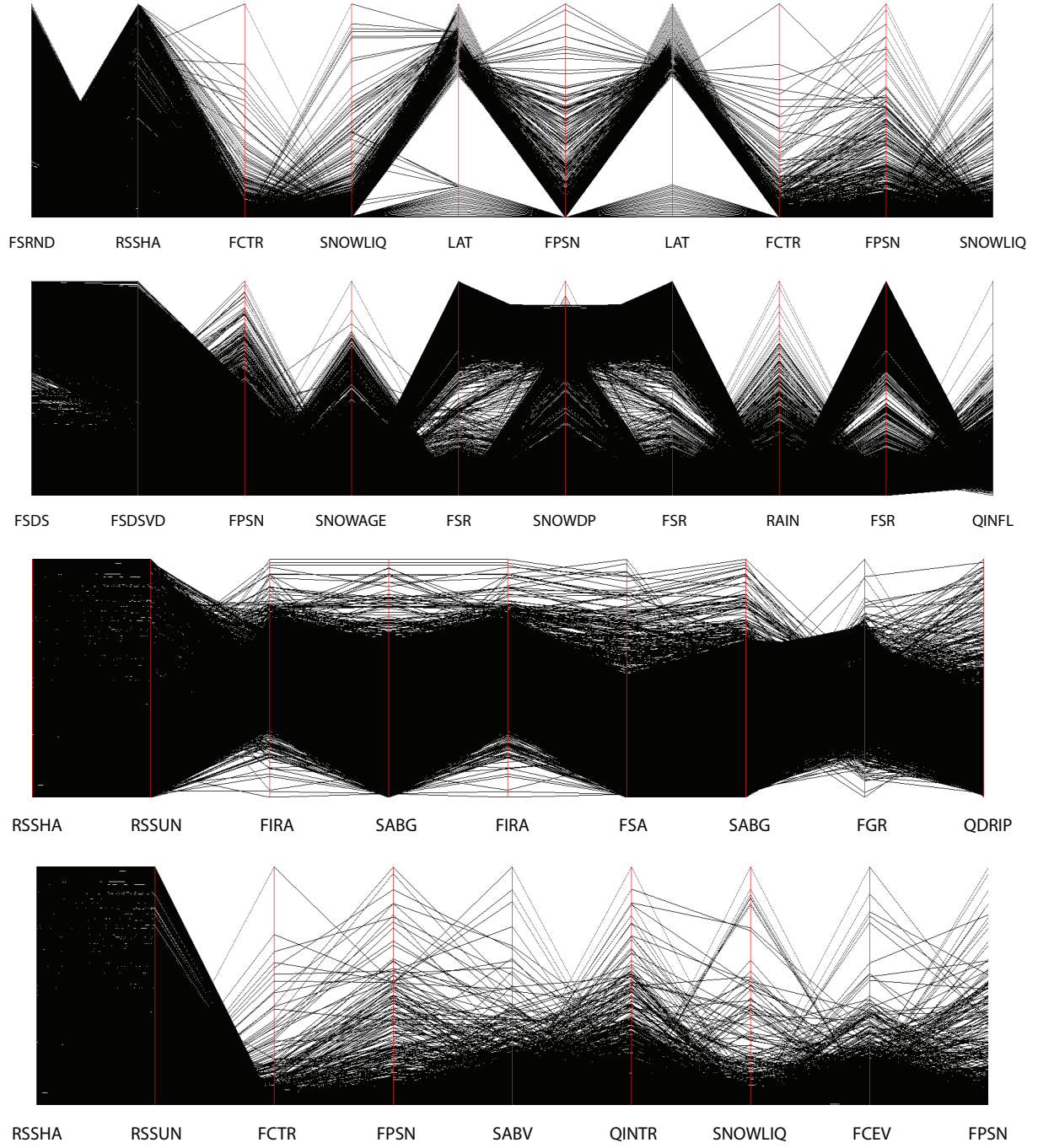


Figure 6.15: Final renderings using decluttering results. Performing directed decluttering on an axis pair moves its location in metric space (to be closer to its target axis pair). The selected axis pairs in these renderings are those closest to this new point.

visual trends, and based on this we allow for quite specific navigation through large numbers of parallel coordinate axis pairs through classification. In addition, we provide a system that acts as a framework for users to create their own sets of metrics, and thereby perform specially tailored classifications. With metrics being leveraged in this capacity, we revisit their role in decluttering and provide a simple but robust directed method for trend detection within cluttered parallel coordinate plots. As a bonus, the results of this approach intrinsically provide feedback to a metric's designer with regard to the accuracy of desired image space measurements.

With this in mind, it is clear from some of the runs of the decluttering routine that there are areas that could be more accurately modeled with new metrics. In the case of our metrics, the decluttering approach illuminates an area for potential improvement, i.e. the location of features within a parallel coordinate plot. In the bottom row of Figure 6.8 it's clear that the fanning effect in the target image has no counterpart in the decluttered S, and it's representation in decluttered S is not nearly as contained as it is in the target. This is another area in which a better metric must be crafted. Also, in our final renderings, we allow for multiple redundant axes. In the future we intend to study whether or not it is beneficial to restrict this, in terms of using the renderings for actual data analysis.

Lastly, there are likely properties which may only be distinguishable with metrics other than those developed in metric space. An example of one such metric is a measure of the average slope of the lines in a parallel coordinate plot. In a future work, we would like to explore some such data space metrics in this framework, and perform an extensive analysis on the data independence of this approach. Also, in the sense that metrics are commonly used to measure "goodness" we would like to explore metrics that measure the performance of our classification metrics.

Chapter 7

Conclusion

In this dissertation, we have presented several elements of successfully exploring the concurrent visualization of multiple multivariate relationships. We first presented a novel multivariate visualization technique for the creation of attribute subspaces and the simultaneous rendering of these subspaces. This summarizing visualization is already useful as we have shown in an application study utilizing only that approach.

Then, we demonstrated a robust framework for the specification and rendering of multivariate relationships. This method is a natural extension of attribute subspaces, in that the subspaces created may now refer to one of a practically limitless number of relationships, as well as simply an attribute. We have shown the utility of being able to quickly specify relationships, even unlikely ones, in terms of both data exploration and in feature discover. These results however, were all in the scope of bivariate relationships. The reason for this, is that even though this method is capable of higher-dimensional relationships, the problem of specifying these types of relationships as well as interpreting the results is very difficult. This is the motivation for the work in Chapter 6, the studying of discovering meaningful high-dimensional relationships.

In that chapter, we presented new work in a field that has been long used for the exact purpose of understanding multivariate relationships, parallel coordinates. Specifically, we outline a user driven approach for selecting parallel coordinate renderings. This selection is based on either a priori knowledge/interest in specific bivariate relationships or in the discovery of new ones. In this final chapter, we outline bringing these multivariate relationships into perspective by providing representative renderings back in original data space. As described in Chapter 5, the input to

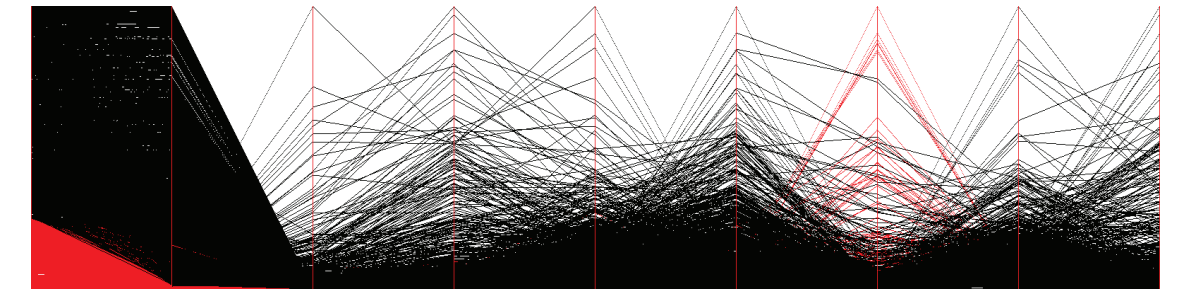
our relationship visualization method is a relationship specification file. We first discuss considerations of the conversion of parallel coordinate renderings to those files, then demonstrate the results of doing so.

For the results and discussions in this chapter, we will be using the rendering results from our parallel coordinate methods (Figures 6.14 and 6.15). Now, each of these parallel coordinate renderings represents a high-dimensional relationship, but in terms of how every spatial point relates to every other over that relationship. In other words, a relationship specification encapsulating every line in the parallel coordinate rendering would guarantee that this relationship would be a perfect fit for every spatial point in the dataset. This would lead to a rendered image with all pixels identically colored.

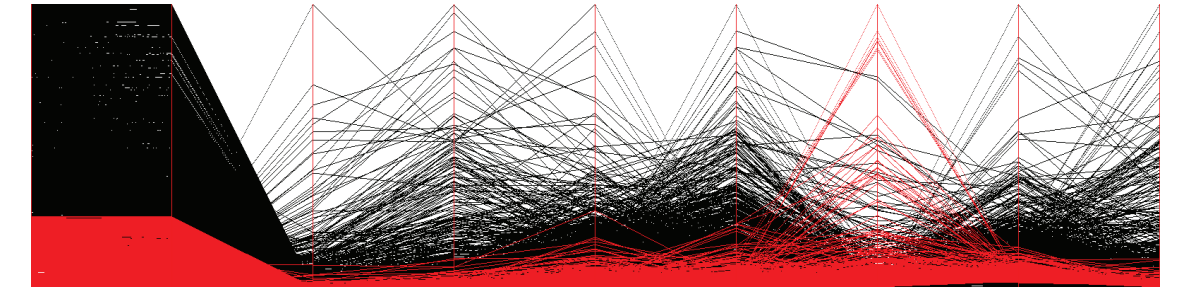
For this reason, a relationship must be some subset of lines, as in Figure 7.1 (a). This relationship is chosen by supplying our augmented renderer with a range of values between [0, 1]. That range corresponds to the start and end of the relationship. The lower and upper bounds are the bottom and top of an axis, respectively. In that figure, we have selected a set of lines corresponding to a section of the parallel coordinate rendering containing the lowest values for the first variable. Also, in that image (a) and (b) illustrate the difference in drawing the relationship before/after the remaining points, respectively. For the rest of the results, the points not fitting a chosen relationship are drawn first (black), then the relationships are drawn in the order they are specified.

Since the visualization approach presented in Chapter 5 employs a best fit competition process, only a subset of the relationship chosen need be represented in the RS-file for an accurate representation of the relationship. For the results generated in this chapter, each relationship is represented by only ten points, uniformly sampled throughout the relationship. Figure 7.1 (c) shows the result on running the relationship visualization approach on the RS-file generated from the relationship shown in Figure 7.1 (a) and (b). This high-dimensional relationship is intuitive, all variables are low, except SNOWLIQ, which has some high values. Indeed, this relationship corresponds directly to the coldest areas of the globe.

In addition to selecting the range of a relationship, we allow for two extra specifications: selection of any axis and multiple relationship specifications. Figure 7.2 demonstrates why this is a desirable aspect to incorporate. In that parallel coordinate rendering, the fifth axis has two clear areas defining separate relationships. With this setup, we can specify each of these as a relationship



(a)



(b)

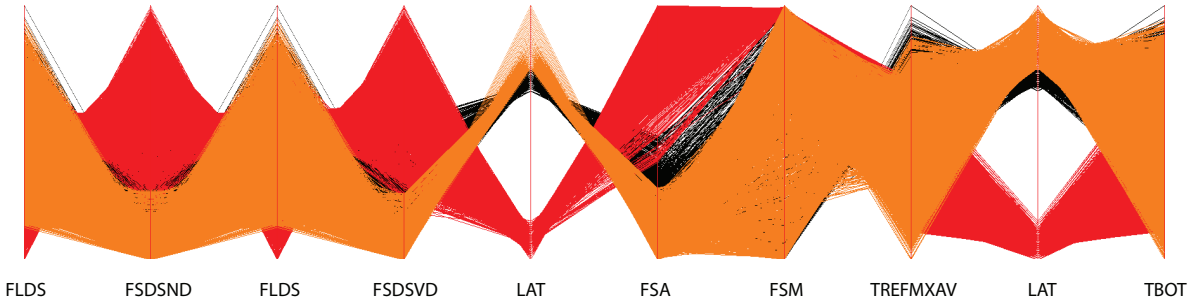
(c)

Figure 7.1: (a) Sample relationship, drawn before remaining lines. (b) Same relationship drawn after. (c) Relationship's representation in data space.

and render both in data space (Figure 7.2(a) and (b)). Also, in the remaining images we show all outputs from the visualization program: the pixels colored simply by relationships, the scores of how well a relationship fits at a point, and a composite of the two. The scores are shown inverted, to ensure better contrast between high scores and empty space.

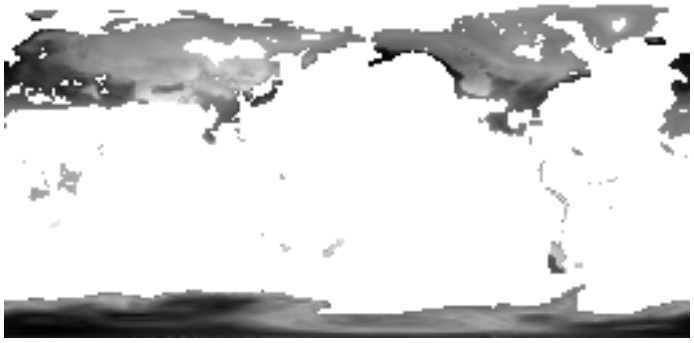
The variable corresponding to the axis chosen in Figure 7.2 is Latitude. The data space renderings indeed show that the two relationships correspond to low and high latitude ranges. For a similar result, we performed selections on Longitude in another parallel coordinate plot. This is shown in Figure 7.3. This image has many interesting implications. First, there are areas of the second relationship that are clearly in the area of the first relationship's longitude range. For this region, the range of the remaining variables are a better fit for the second relationship, enough that the actual longitude values are negligible. However, the scores of the fit make it clear that this region is not perfectly well fit by either relationship. Also, the line where the two relationships meet is clearly marked in the data space images, and is skewed quite far from the middle, illustrating exactly how normalization effected this variable.

Finally, we take all remaining parallel coordinate renderings we have not yet looked at from Figures 6.14 and 6.15 and create one rendering in data space of eight separate relationships, two from each of these four parallel coordinate renderings. These four parallel coordinate renderings are shown in Figure 7.4, and the resulting data space rendering is shown in Figure 7.5. In this rendering, there are 21 different variables represented in at least one relationship, with the relationships being between five- and eight-dimensional.

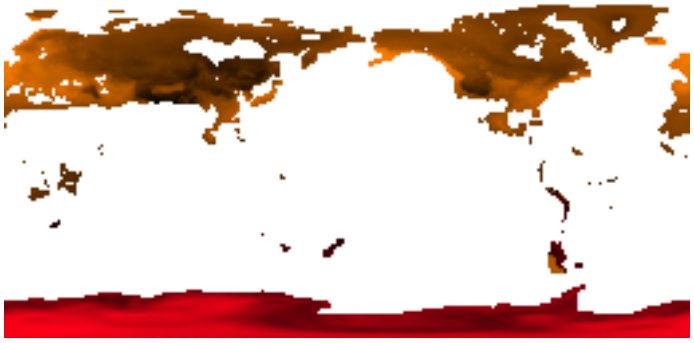


(a)

(b)

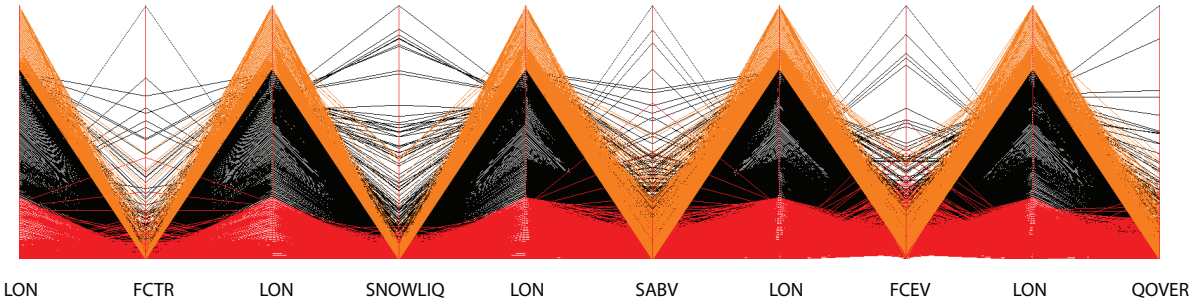


(c)



(d)

Figure 7.2: (a) Two relationships chosen on Latitude axis. (b) Relationships in data space. (c) Scores of relationship fits. (d) Composite of (b) and (c).



(a)

(b)

(c)

(d)

Figure 7.3: (a) Two relationships chosen on Longitude axis. (b) Relationships in data space. (c) Scores of relationship fits. (d) Composite of (b) and (c).

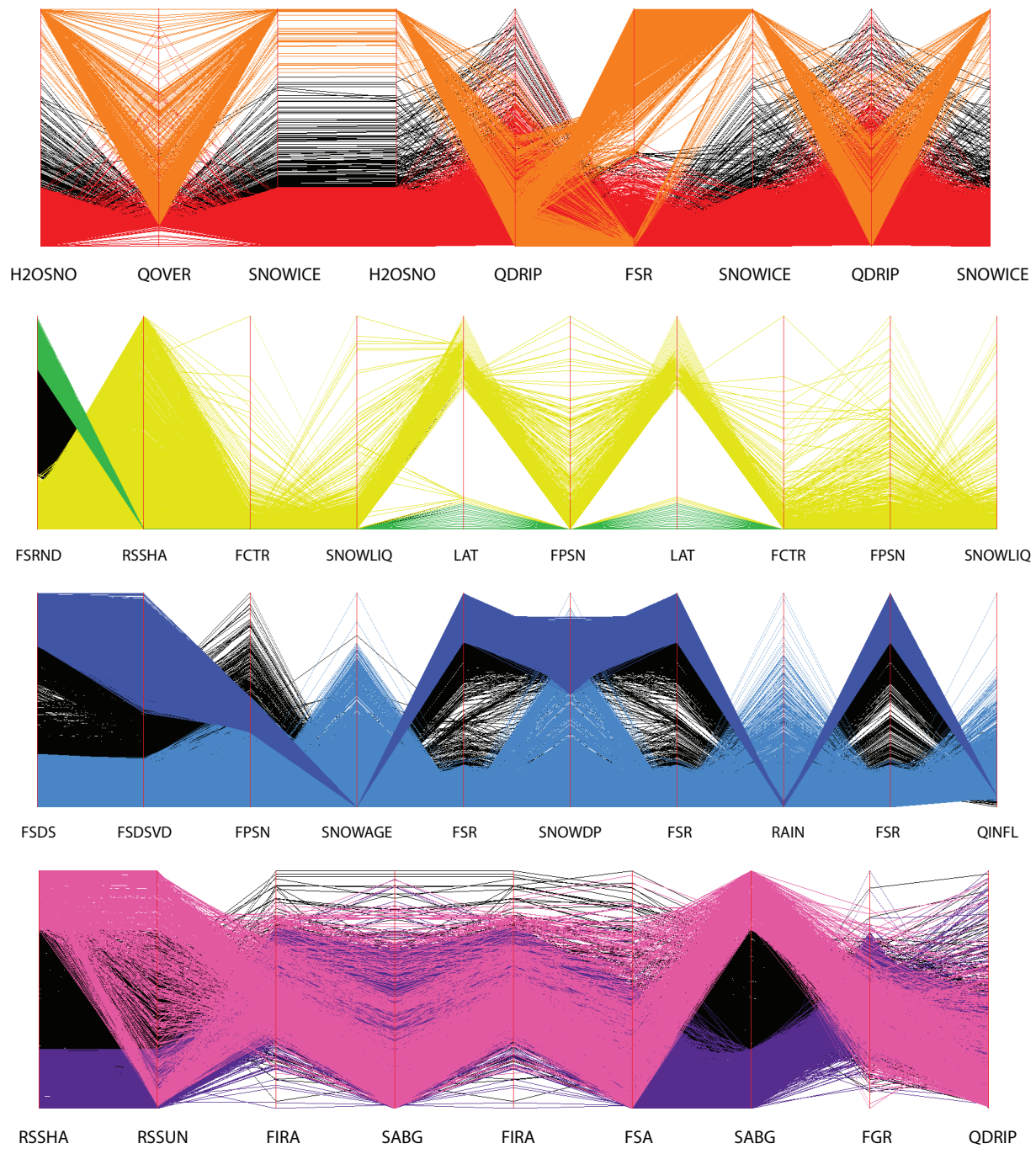


Figure 7.4: Parallel coordinate renderings of the eight relationships for the data space rendering in Figure 7.5.

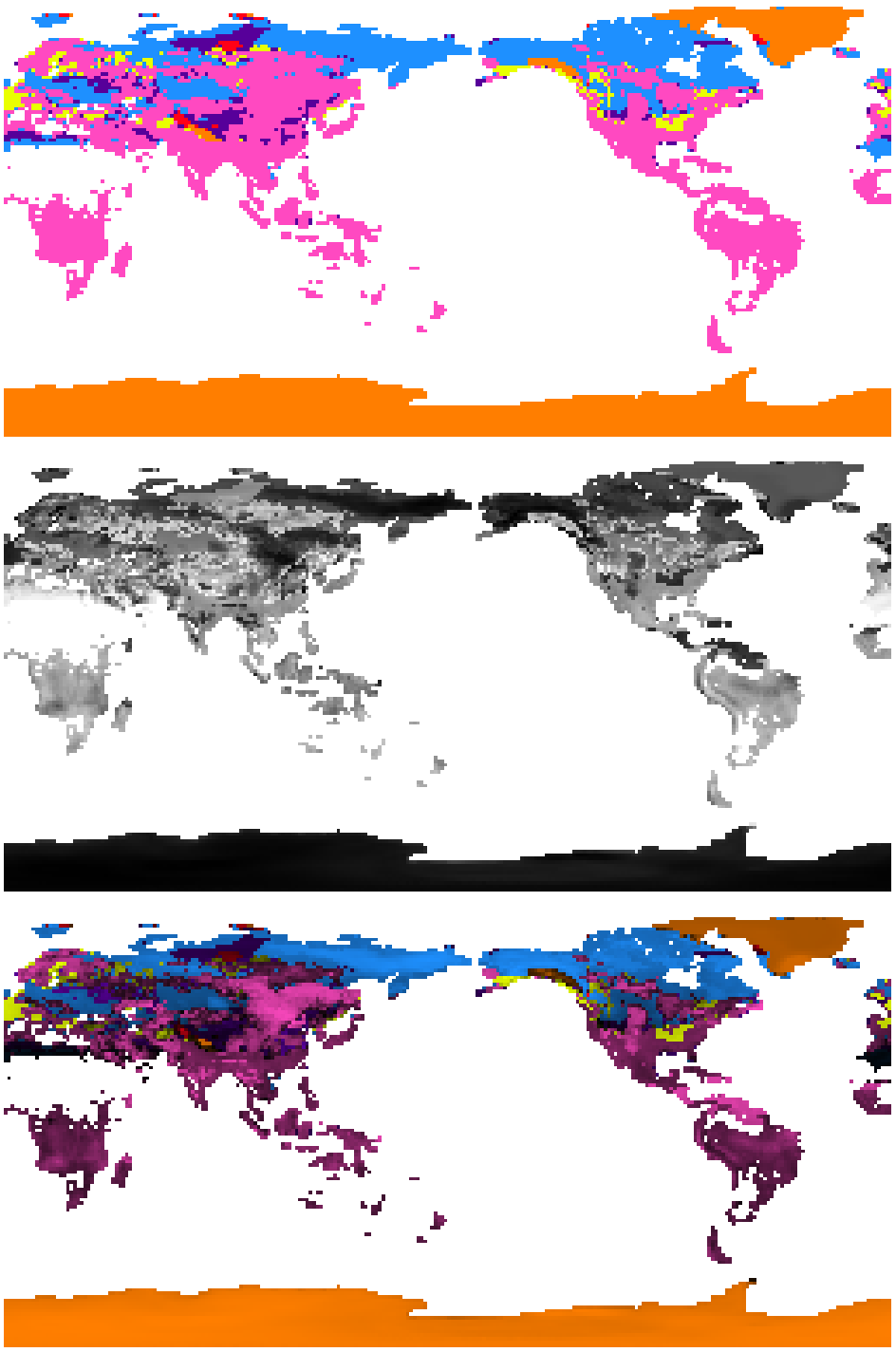


Figure 7.5: Data space renderings of the relationships in Figure 7.4

Bibliography

Bibliography

[Ahlberg, 1996] Ahlberg, C. (1996). Spotfire: an information exploration environment. *SIGMOD Rec.*, 25(4):25–29.

[Ahlberg and Shneiderman, 1994] Ahlberg, C. and Shneiderman, B. (1994). Visual information seeking: tight coupling of dynamic query filters with starfield displays. In *CHI '94: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 313–317, New York, NY, USA. ACM Press.

[Ahlberg et al., 1992] Ahlberg, C., Williamson, C., and Shneiderman, B. (1992). Dynamic queries for information exploration: an implementation and evaluation. In *CHI '92: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 619–626, New York, NY, USA. ACM Press.

[Bair et al., 2006] Bair, A., House, D. H., and Ware, C. (2006). Texturing of layered surfaces for optimal viewing. *IEEE Trans. Vis. Comput. Graph.*, 12(5):1125–1132.

[Beddow, 1990] Beddow, J. (1990). Shape coding of multidimensional data on a microcomputer display. In *Proceedings of IEEE Visualization 90*, pages 238–246.

[Brodbeck et al., 1997] Brodbeck, D., Chalmers, M., Lunzer, A., and Cotture, P. (1997). Domesticating bead: Adapting an information visualization system to a financial institution. In *Proc. IEEE Information Visualization 97*, pages 73–80. IEEE Computer Society Press.

[Derthick et al., 2003] Derthick, M., Christel, M. G., Hauptmann, E. G., and Wactlar, H. D. (2003). Constant density displays using diversity sampling.

[DoE, 2000] DoE, U. S. (2000). Scientific discovery through advanced scientific computing.

[dos Santos and Brodlie, 2004] dos Santos, S. and Brodlie, K. (2004). Gaining understanding of multivariate and multidimensional data through visualization. *Computers and Graphics*, 28(3):311–325.

[Draper et al., 2008] Draper, G., Livnat, Y., and Riesenfeld, R. (2008). A visual query language for correlation discovery and management. In *Proceedings of Visual and Iconic Language Conference (VaIL 2008)*, pages 14–23.

[Drebin et al., 1988] Drebin, R. A., Carpenter, L., and Hanrahan, P. (1988). Volume rendering. In *Computer Graphics (Proceedings of SIGGRAPH 88)*, volume 22, pages 65–74.

[Ellis et al., 2005] Ellis, G., Bertini, E., and Dix, A. (2005). The sampling lens: making sense of saturated visualisations. In *CHI '05: CHI '05 extended abstracts on Human factors in computing systems*, pages 1351–1354, New York, NY, USA. ACM.

[Ellis and Dix, 2006] Ellis, G. and Dix, A. (2006). Enabling automatic clutter reduction in parallel coordinate plots. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):717–724.

[Ellis and Dix, 2007] Ellis, G. and Dix, A. (2007). A taxonomy of clutter reduction for information visualisation. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1216–1223.

[Enrico Bertini, 2006] Enrico Bertini, L. Dell'Aquila, G. S. (2006). Reducing infovis cluttering through non uniform sampling, displacement, and user perception.

[Ferreira and Levkowitz, 2003] Ferreira, M. C. and Levkowitz, H. (2003). From visual data exploration to visual data mining: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 09(3):378–394.

[Fua et al., 1999] Fua, Y. H., Ward, M. O., and Rundensteiner, E. A. (1999). Hierarchical parallel coordinates for exploration of large datasets. In *Proceedings of IEEE Visualization '99*, pages 43–50.

[F.Vega et al., 2005] F.Vega, Hastreiter, P., Naraghi, R., Fahlbusch, R., and Greiner, G. (2005). Smooth volume rendering of labeled medical data on consumer graphics hardware. In *Proceedings of SPIE Medical Imaging 2005*.

[Glatter et al., 2006] Glatter, M., Mollenhour, C., Huang, J., and Gao, J. (2006). Scalable data servers for large multivariate volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1291–1298.

[Gosink et al., 2007] Gosink, L., Anderson, J., Bethel, W., and Joy, K. (2007). Variable interactions in query-driven visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1400–1407.

[Graham and Kennedy, 2003] Graham, M. and Kennedy, J. (2003). Using curves to enhance parallel coordinate visualisations. In *IV '03: Proceedings of the Seventh International Conference on Information Visualization*, pages 10–16.

[Healey and Enns, 2002] Healey, C. G. and Enns, J. T. (2002). Perception and Painting: A Search for Effective, Engaging Visualizations. *IEEE Computer Graphics and Applications*, 22(2):10–15.

[Helgeland and Andreassen, 2004] Helgeland, A. and Andreassen, O. (2004). Visualization of vector fields using seed lic and volume rendering. *IEEE Trans. Vis. Comput. Graph.*, 10(6):673–682.

[Hoffman et al., 2007] Hoffman, F., Covey, C., Fung, I., Randerson, J., Thornton, P., Lee, Y.-H., Rosenbloom, N., Stockli, R., Running, S., Bernholdt, D. E., and Williams, D. (2007). Results from the carbon-land model intercomparison project (c-lamp) and availability of the data on the earth system grid (esg). *Journal of Physics: Conference Series*, 78:012026.

[Hoffman et al., 2005] Hoffman, F., Hargrove, W. W., Erickson, D. J., and Oglesby, R. J. (2005). Using clustered climate regimes to analyze and compare predictions from fully coupled general circulation models. *Earth Interactions*, 9(10):1–27.

[Inselberg, 1985] Inselberg, A. (1985). The plane with parallel coordinates. *The Visual Computer*, 1(2):69–91.

[Inselberg and Dimsdale, 1990] Inselberg, A. and Dimsdale, B. (1990). Parallel coordinates: A tool for visualizing multi- dimensional geometry. In *Proceedings of IEEE Visualization 90*, pages 361–375.

[Inselberg and Dimsdale, 1994] Inselberg, A. and Dimsdale, B. (1994). Multidimensional lines i: Representation. *SIAM J. Appl. Math.*, 54(2):559–577.

[Jänicke et al., 2007] Jänicke, H., Wiebel, A., Scheuermann, G., and Kollmann, W. (2007). Multifield visualization using local statistical complexity. *IEEE Trans. Vis. Comput. Graph.*, 13(6):1384–1391.

[Johansson et al., 2005a] Johansson, J., Cooper, M., and Jern, M. (2005a). 3-dimensional display for clustered multi-relational parallel coordinates. In *IV '05: Proceedings of the Ninth International Conference on Information Visualisation*, pages 188–193.

[Johansson et al., 2008] Johansson, j., Forsell, C., Lind, M., and Cooper, M. (2008). Perceiving patterns in parallel coordinates: determining thresholds for identification of relationships. *Information Visualization*, 7(2):152–162.

[Johansson et al., 2005b] Johansson, J., Ljung, P., Jern, M., and Cooper, M. (2005b). Revealing structure within clustered parallel coordinates displays. In *INFOVIS '05: Proceedings of the Proceedings of the 2005 IEEE Symposium on Information Visualization*, page 17.

[Kindlmann and Durkin, 1998] Kindlmann, G. and Durkin, J. W. (1998). Semi-automatic generation of transfer functions for direct volume rendering. In *IEEE Symposium on Volume Visualization*, pages 79–86.

[Kindlmann et al., 2003] Kindlmann, G., Whitaker, R., Tasdizen, T., and Möller, T. (2003). Curvature-based transfer functions for direct volume rendering: Methods and applications. In *Proceedings of IEEE Visualization '03*, pages 513–520.

[Kirby et al., 1999] Kirby, R. M., Marmanis, H., and Laidlaw, D. H. (1999). Visualizing multivalued data from 2D incompressible flows using concepts from painting. In Ebert, D., Gross, M., and Hamann, B., editors, *IEEE Visualization '99*, pages 333–340, San Francisco.

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

[Kniss et al., 2001] Kniss, J., McCormick, P., McPherson, A., Ahrens, J., Painter, J., Keahey, A., and Hansen, C. (2001). Interactive texture-based volume rendering for large data sets. *IEEE Computer Graphics and Applications*, 21(4):52–61.

[LeBlanc et al., 1990] LeBlanc, J., Ward, M. O., and Wittels, N. (1990). Exploring n-dimensional databases. In *Proceedings of IEEE Visualization 90*, pages 230–237.

[Levkowitz, 1991] Levkowitz, H. (1991). Color icons: Merging color and texture perception for integrated visualization of multiple parameters. In *Proceedings of IEEE Visualization 91*, pages 164–170.

[Love et al., 2005] Love, A. L., Pang, A., and Kao, D. L. (2005). Visualizing spatial multivalue data. *IEEE Computer Graphics and Applications*, 25(3):69–79.

[Lu et al., 2007] Lu, A., Ebert, D. S., Qiao, W., Kraus, M., and Mora, B. (2007). Volume illustration using wang cubes. *ACM Trans. Graph.*, 26(2).

[Lu et al., 2003] Lu, A., Morris, C. J., Taylor, J., Ebert, D. S., Hansen, C., Rheingans, P., and Hartner, M. (2003). Illustrative interactive stipple rendering. *IEEE Transactions on Visualization and Computer Graphics*, 9(2):127 – 138.

[McCormick et al., 2004] McCormick, P., Inman, J., Ahrens, J., Hansen, C., and Roth, G. (2004). Scout: A hardware-accelerated system for quantitatively driven visualization and analysis. In *Proceedings of IEEE Visualization '04*, pages 171–178.

[McDonnell and Mueller, 2008] McDonnell, K. T. and Mueller, K. (2008). Illustrative parallel coordinates. *Comput. Graph. Forum*, 27(3):1031–1038.

[Mihalisin et al., 1991a] Mihalisin, T., Timlin, J., and Schwegler, J. (1991a). Visualization and analysis of multi-variate data: A technique for all fields. In *Proceedings of IEEE Visualization 91*, pages 171–178.

[Mihalisin et al., 1991b] Mihalisin, T., Timlin, J., and Schwegler, J. (1991b). Visualizing multivariate functions, data, and distributions. *IEEE Computer Graphics and Applications*, 11(3):28–35.

[Moustafa and Wegman, 2002] Moustafa, R. E. A. and Wegman, E. J. (2002). On some generalizations of parallel coordinate plots. In *Seeing a Million: A Data Visualization Workshop*.

[Neumann et al., 2000] Neumann, L., Csébfalvi, B., König, A., and Gröller, E. (2000). Gradient estimation in volume data using 4D linear regression. In Gross, M. and Hopgood, F. R. A., editors, *Computer Graphics Forum (Eurographics 2000)*, volume 19, pages 351–358.

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

[Pascucci and Frank, 2001] Pascucci, V. and Frank, R. J. (2001). Global static indexing for real-time exploration of very large regular grids. In *Supercomputing '01: Proceedings of the 2001 ACM/IEEE conference on Supercomputing (CDROM)*, pages 2–2, New York, NY, USA. ACM.

[Peng et al., 2004] Peng, W., Ward, M. O., and Rundensteiner, E. A. (2004). Clutter reduction in multi-dimensional data visualization using dimension reordering. In *In INFOVIS 04: Proceedings of the IEEE Symposium on Information Visualization (INFOVIS04*, pages 89–96. IEEE Computer Society.

[Rafiei and Curial, 2005] Rafiei, D. and Curial, S. (2005). Effectively visualizing large networks through sampling. In *IEEE Visualization*, page 48.

[Riley et al., 2003] Riley, K., Ebert, D., Hansen, C., and Levit, J. (2003). Visually accurate multi-field weather visualization. In *IEEE Visualization '03*, pages 279 – 286.

[Sauber et al., 2006] Sauber, N., Theisel, H., and Seidel, H.-P. (2006). Multifield-graphs: An approach to visualizing correlations in multifield scalar data. *IEEE Trans. Vis. Comput. Graph.*, 12(5):917–924.

[Steed et al., 2007] Steed, C. A., Fitzpatrick, P. J., Jankun-Kelly, T. J., and Yancey, A. N. (2007). Practical application of parallel coordinates to hurricane trend analysis. In *Proceedings of IEEE Visualization'07*.

[Stockinger et al., 2005] Stockinger, K., Shalf, J., Bethel, W., and Wu, K. (2005). Query driven visualization of large data sets. In *Proceedings of IEEE Visualization '05*, pages 167–174.

[Taylor, 2002] Taylor, R. (2002). Visualizing multiple fields on the same surface. *IEEE Computer Graphics and Applications*, 22(3):6–10.

[Tiede et al., 1998] Tiede, U., Schiemann, T., and Höhne, K. H. (1998). High quality rendering of attributed volume data. In Ebert, D., Hagen, H., and Rushmeier, H., editors, *IEEE Visualization '98*, pages 255–262.

[Tzeng et al., 2003] Tzeng, F.-Y., Lum, E., and Ma, K.-L. (2003). A novel interface for higher-dimensional classification of volume data. In *Proceedings of IEEE Visualization '03*, pages 505–512.

[Viola et al., 2004] Viola, I., Kanitsar, A., and Gröller, M. E. (2004). Importance-driven volume rendering. In *Proceedings of IEEE Visualization '04*, pages 139–145.

[Walter and Healey, 2001] Walter, J. D. and Healey, C. G. (2001). Attribute preserving dataset simplification. In *IEEE Visualization 2001*, pages 113–120.

[Wegman and Luo,] Wegman, E. J. and Luo, Q. High dimensional clustering using parallel coordinates and the grand tour. *Computing Science and Statistics*, (124).

[Wegman, 1990] Wegman, J. E. (1990). Hyperdimensional data analysis using parallel coordinates. *J. Am. Stat. Assn*, 85(411):664–675.

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

[Wong and Bergeron, 1997a] Wong, P. and Bergeron, R. (1997a). 30 years of multidimensional multivariate visualization. In *Scientific Visualization - Overviews, Methodologies and Techniques*, pages 3 – 33.

[Wong and Bergeron, 1997b] Wong, P. C. and Bergeron, R. D. (1997b). Multivariate visualization using metric scaling. In *VIS '97: Proceedings of the 8th conference on Visualization '97*, pages 111–ff., Los Alamitos, CA, USA. IEEE Computer Society Press.

[Wong et al., 2002] Wong, P. C., Foote, H., Kao, D. L., Leung, R., and Thomas, J. (2002). Multivariate visualization with data fusion. *Information Visualization*, 1(3/4):182–193.

[Wong et al., 1999] Wong, P. C., Whitney, P., and Thomas, J. (1999). Visualizing association rules for text mining. In *INFOVIS '99: Proceedings of the 1999 IEEE Symposium on Information Visualization*, page 120, Washington, DC, USA. IEEE Computer Society.

[Woodring and Shen, 2003] Woodring, J. and Shen, H. (2003). Chronovolumes: A direct rendering technique for visualization time-varying data. In *Eurographics/IEEE TVCG Workshop on Volume Graphics*, pages 27 – 34.

[Woodring and Shen, 2006] Woodring, J. and Shen, H.-W. (2006). Multi-variate, time varying and comparative visualization with contextual cues. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):909–916.

[Woodring et al., 2003] Woodring, J., Wang, C., and Shen, H.-W. (2003). High dimensional direct rendering of time-varying volumetric data. In *Proceedings of IEEE Visualization '03*.

[Zhou et al., 2008] Zhou, H., Yuan, X., Qu, H., Cui, W., and Chen, B. (2008). Visual clustering in parallel coordinates. *Computer Graphics Forum*, 27(3):1047–1054.

Vita

Robert Sisneros was born April 15, 1982 and raised in Clarksville, TN. In Fall 2003, he completed undergraduate degrees in Mathematics and Computer Science at Austin Peay State University. He subsequently was admitted as a graduate assistant at the University of Tennessee in Knoxville. There, he started work with Dr. Jian Huang in Spring 2004 and received his M.S. in computer science in Fall 2005. In Spring 2009 he earned the Ph.D. degree in computer science, specializing in Scientific Visualization.