TARGETING DESIGNS OF SCALABLE, EXPLORATORY SUMMARY VISUALIZATIONS

by

Alper T. Sarikaya

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

(Computer Sciences)

at the

UNIVERSITY OF WISCONSIN–MADISON

2017

Date of final oral examination: 07/25/2017

The dissertation is approved by the following members of the Final Oral Committee:
Michael Gleicher (Chair), Professor, Computer Sciences
Bilge Mutlu, Associate Professor, Computer Sciences
Eftychios Sifakis, Assistant Professor, Computer Sciences
Robert Roth, Associate Professor, Geography
Danyel Fisher, Senior Researcher, Microsoft Research

For the sake of humankind to satisfy its never-ending thirst for insight.

ACKNOWLEDGMENTS

My educational journey could not have been possible without the support of many enthusiastic mentors. I would like to thank my committee for their feedback and guidance over the course of my graduate work, particularly for pointing out those seemingly little tidbits that radically changed my thinking. I am indebted to them for my growth as a researcher, and I hope to support others to realize their potential as they have supported me. Special thanks are warranted for my advisor, Prof. Michael Gleicher for his advisement and encouragement to perform high-quality work, regardless of the circumstances.

The area of data visualization is a very application-driven field, and I am thankful for the many fruitful collaborations over the course of my studies. Prof. George Phillips and Prof. Julie Mitchell were instrumental in forming the problem characterization in Chapter 5 (with help from student Spencer Eriksen), and I had the great pleasure of working with Prof. David O'Connor, Prof. Thomas Friedrich, and student Jorge Dinis on the virology work in Chapter 6. All had the patience to explain their science to a computer science student and generously participated in the design iteration of problem-specific visualization.

My time in Madison has been greatly enriched by those students surrounding me in the Visual Computing Lab, and I thank them for productive discussion, critical feedback, and the chance to land a hit in Super Smash Bros. 64 during free moments. The visualization crew of Eric Alexander, Chih-Ching Chang, Michael Correll (Chapter 6), Adrian Mayorga (Chapter 7), and Danielle Szafir (Chapters 3 & 5) all helped to broaden my understanding of visualization and its wide-ranging applications. The physical simulation crew of Michael Doescher, Haixiang Liu, Nathan Mitchell, the animation/robot crew of Sean Andrist, Tomislav Pejsa, Danny Rakita, and Chris Bodden, and resident owl Brandon Smith all contributed academically and socially to the work presented herein. Apologies to Shaquille O'Neal, who was left off this list.

Sarcastic thanks are in order for the easily-bikable city of Madison, Wisconsin, without which I probably would have finished my studies a year earlier. Sincere thanks to JJ, Barrett, and the entire Indie Coffee staff, who provided a warm environment for over half of my writing in graduate school.

I want to thank the people in my life that have always supported me in my educational endeavors. My parents Demet and Mehmet have always fought for and positioned me to have the best educational opportunities, and I can only hope to be as voracious and compassionate as them. My friends in Madison have most definitely helped me grow as an individual, and I thank Abby, Alexis, Bob, Brad, Collin, Jason, JJ, Mabel, Scrantz, Shanda, and Tiffany for their companionship, laughter, and support. There are no words to capture the impact of my wife Amanda, who is one of the most compassionate, considerate, and pragmatic people I know. I look forward to attacking the problems of the world and enabling others to achieve their fullest potential with her in the near future.

Work in this thesis was supported by grants from the National Science Foundation (CMMI-0941013 and IIS-1162037), National Institutes of Health (award 5R01AI077376-07), the Department of Computer Sciences (Summer 2012), and the Andrew W. Melon Foundation. Specifically, Chapters 3–7 were supported by IIS-1162037, Chapter 5 by CMMI-0941013, Chapters 5–7 by 5R01AI077376-07.

CONTENTS

Contents iv

List of Tables vi

List of Figures vii

Abstract ix

1 Introduction 1

1.1 Definitions, Background, and Basis 2

1.2 Addressing the Need for Guidelines 6

1.3 Document Structure 8

2 Background 10

2.1 Summarizing Data in Visualization 10

2.2 Taxonomies and Organization of the Visualization Design Space 13

2.3 Abstractions for the Visualization Designer 17

3 Design Factors for Summary Visualization 25

3.1 Background 27

3.2 Methodology 31

3.3 Survey Results 37

3.4 Discussion 50

4 Design Factors for Scatterplots at Scale 57

4.1 Overview 57

4.2 Background 58

4.3 Scatterplot Tasks 63

4.4 Data Characteristics 67

4.5 Design Decisions 71

4.6 Using the Framework to Drive Design 77

4.7 Discussion 83

4.8 Conclusion 87

5 Case Study: Visualizing Validation for Protein Surface Classifiers 89

5.1 Overview 90

5.2 Background 93

5.3 Experiment Overviews 95

5.4 Detail View 99

5.5 Use Cases 104

5.6 Discussion 108

6 Case Study: Visualizing Co-occurrence in Populations of Viral Genomes 110

6.1 Overview 110

6.2 Biological Background 112

6.3 Problem Details and Requirements 114

6.4 Related Work 116

6.5 Interest Metrics 118

6.6 Visualization Design 121

6.7 Case Studies 128

6.8 Discussion 131

7 Programming Abstractions of Scalable Visualization 134

7.1 Using WebGL: Implementing the Splatterplot System 134

7.2 `d3-twodim`*: Scatterplot-like Designs in the Browser* 145

7.3 Discussion 147

8 Discussion 149

8.1 Utility of the Proposed Framework 150

8.2 Future Work 151

Bibliography 156

LIST OF TABLES

3.1 Codebook for the QCA survey of summary visualizations 34
3.2 Themes found in analytics tools using summary visualizations 38

4.1 The 12 collected and abstrracted analysis tasks that are performed with scatterplots 65
4.2 The data attributes that affect scatterplot design 68
4.3 Clustering of design decisions for scatterplots 74
4.4 A 2D slice of the task support map, highlighting task support by design decisions 80

LIST OF FIGURES

1.1 Example: abstracting data in different ways will surface different information 3
1.2 Summary visualization in the wild: Summarizing the performance of a protein structural classifier 5

2.1 Norman's Action Model 18
2.2 Example: different data have the same statistics 21

3.1 Teaser: A schematic of a generalized process of creating summary visualizations 26
3.2 Distributions of purpose for the analyzed summary visualizations 39
3.3 Example: The system built by Chen et al. [2016] supports a wide range of high-level tasks 40
3.4 Example: World Lines uses aggregation to simplify searching for motifs . . . 41
3.5 Distribution of data summarization methods for the analyzed summary visualizations 42
3.6 Example: Many network summaries combine aggregation and filtering 43
3.7 Example: Splatterplots combine aggregation with filtering and subsmapling . 44
3.8 Distribution of tasks for the analyzed summary visualizations 47
3.9 Example: Summaries supporting browsing highlight distributions and clusters 48
3.10 Example: Summaries often act as roadmaps for exploration 48

4.1 Teaser: effective scatterplot design depends on the task and data chacateristics 59
4.2 Example: Distributions of data that can prompt different designs 71
4.3 Example: Comparison of task support between scatterplot designs 85

5.1 Teaser: Visualizing validation of a protein surface classifier experiment 91
5.2 Different glyph encodings for overviews afford different observations about the data. 98
5.3 Clustering similar values creates discrete regions that can be identified visually and by interaction. 100
5.4 Multivariate encoding layers input features and output classification together 102

5.5 Our approach for visualizing a DNA protein surface classifier experiment . . 103
5.6 Analyzing the spatial clustering of a DNA-binding classifier reveals high-level trends of classification. 105
5.7 Analysis of a surface descriptor-based, calcium-binding classifier 107

6.1 Example: An abstraction of viral genomic data 115
6.2 Our initial (negative) prototype to identify pairwise correlations in genomes . 121
6.3 Our initial prototype with a sparse display after filtering 123
6.4 A overview of SIV loaded into *CooccurViewer* 125
6.5 Detail view of a single co-occurrence summary 126
6.6 Case Study: A sample of an H5N1 viral population 129
6.7 Case Study: A sample of a SIV viral population 131

7.1 Our WebGL implementation of the Splatterplot technique, *SplatterJs* 135
7.2 Data flow and rendering steps in the *SplatterJs* system 139
7.3 Example: A year of FARS data in *SplatterJs* 143
7.4 An example of highlighting in `d3-twodim` 146

ABSTRACT

Data visualization provides a human-digestible interface to digital data. With increasing data volumes and increased complexity and interrelationships, so have the demands on supporting effective visualization to support this interface. With complex data at scale, it becomes necessary to summarize the data in some manner to communicate high-level information of a dataset, such as distributions, trends, or anomalies. By the nature of summarization, fidelity in the visual representation of such summaries is reduced. With summarized data, visualization designs must make trade-offs to support particular types of tasks and analyses over others. In this dissertation, I present organizations and applications for the effective design of summary visualizations. Organizations of summary visualization identify the relevant factors that affect appropriate design, such as the method of summarizing data, the analysis goals and tasks of the viewer, and the characteristics of the data. Applications of summary visualizations demonstrate the holistic application of appropriate design decisions to support the analysis of complex scientific data. These factors are linked together to identify appropriate design strategies and highlight open problems for the effective design of visualization. Through this research presented herein, I provide new guidance for effective visualization of collections of data, allowing for the wider dissemination and analysis of complex data.

1 INTRODUCTION

This dissertation provides a framework to guide the effective design of visualizations that **summarize** data. A summary visualzation involves an explicit, conscious choice on behalf of the designer to reduce the amount of data for the purposes of simplifying and focusing the visual result. With increasing data volumes and complexity, visualizations must also be similarly be scalable or risk providing inaccurate or incomplete **information** (representations of the data, such as trends, distribution, or anomalies). The design of such visualizations is often done in an on-demand, ad-hoc, iterative fashion, which builds upon stakeholder feedback and technical considerations [Sedlmair et al., 2012a]. While many typologies and taxonomies exist in the visualization research literature that can provide guidance for making appropriate design decisions, they are often at the wrong level to offer practical options to practitioners, or are specific to a particular type of analysis or data. The work in this dissertation strives to close this gap by identifying trends used in existing summary visualizations—codifying and verifying the methods for minimizing data for summarization and correlating with the type of information conveyed by a resulting visualization. A framework for designing effective summary visualizations can be built by concretizing concepts such as the method of minimizing data, and linking these concepts with what information a viewer[1] wants to obtain from a given dataset.

Such a model for effective summarization does not currently exist. Much of the difficulty of deriving such a model lies that it is difficult to accurately and concisely discuss effective design in the abstract, especially when considering many different circumstances. These different circumstances involve the purpose of the visualization, the types and sequences of analyses performed using the visualization, and the characteristics of the data. Each of the categorizations must be individually understood, organized, and verified. In this dissertation, I describe how previous work can help develop guidelines for summary design, driven by empirical study of the state-of-the-art summary visualizations, a case study for task-appropriate design for scatterplots, and practical applications of these design guidelines for specific, analysis problems dealing with scientific data.

[1]The term "viewer" is used to describe the generalized user or analyst of a visualization in this document.

1.1 Definitions, Background, and Basis

Computer science uses **abstraction** to help discuss design decisions throughout the field. The concept of abstraction helps to manage complexity—abstraction in the general sense helps to derive general concepts from specific examples. Computer science uses abstraction to extract processes and remove specific implementation (and other out-of-scope) details, resulting in models and organizations that can be reused or motivate for new analysis scenarios. To construct this framework for summary in data visualization, abstraction is used to understand the inputs and the visual output of an instantiated visualization. Abstraction in this sense is key to generalize guidelines for effective visualization design. Abstraction is used to organize effective design decisions for design in the data visualization literature. Due to the heterogeneity of the data visualization community, many different abstractions have been made to organize the design and use of visualization for analysis. The data visualization community involves members from perceptual psychology, human-computer interaction, cartography, and computer graphics, and therefore the discourse reflects the values from these communities. These varying levels and avenues of abstraction help to organize the discussion around sub-problems. Regardless of this heterogeneity, there is consensus about some abstractions that are common to all data visualizations: **tasks** and **data characteristics** [Munzner, 2014].

The abstraction of task helps to facilitate and standardize how visualizations are used to generate insight on behalf of the user by avoiding domain-specific terminology. Munzner [2014] describes these abstract tasks as a compound statement, describing the tasks as an action performed on one or more targets—a structure echoed by many task taxonomies in the visualization literature. Tasks describe what aspects of the data need to be visualized, such as trends of all data, particular anomalies or features, one or more data attributes, correlations and trends, or data-specific structures. By understanding the needs of the viewer, a visualization can be deemed as appropriate if it supports the completion of a viewer's task. Tasks are therefore a critical abstraction to help guide the design and construction of visualizations that summarize a large amount of data.

Task abstraction can also illustrate how a visualization's design governs how a viewer

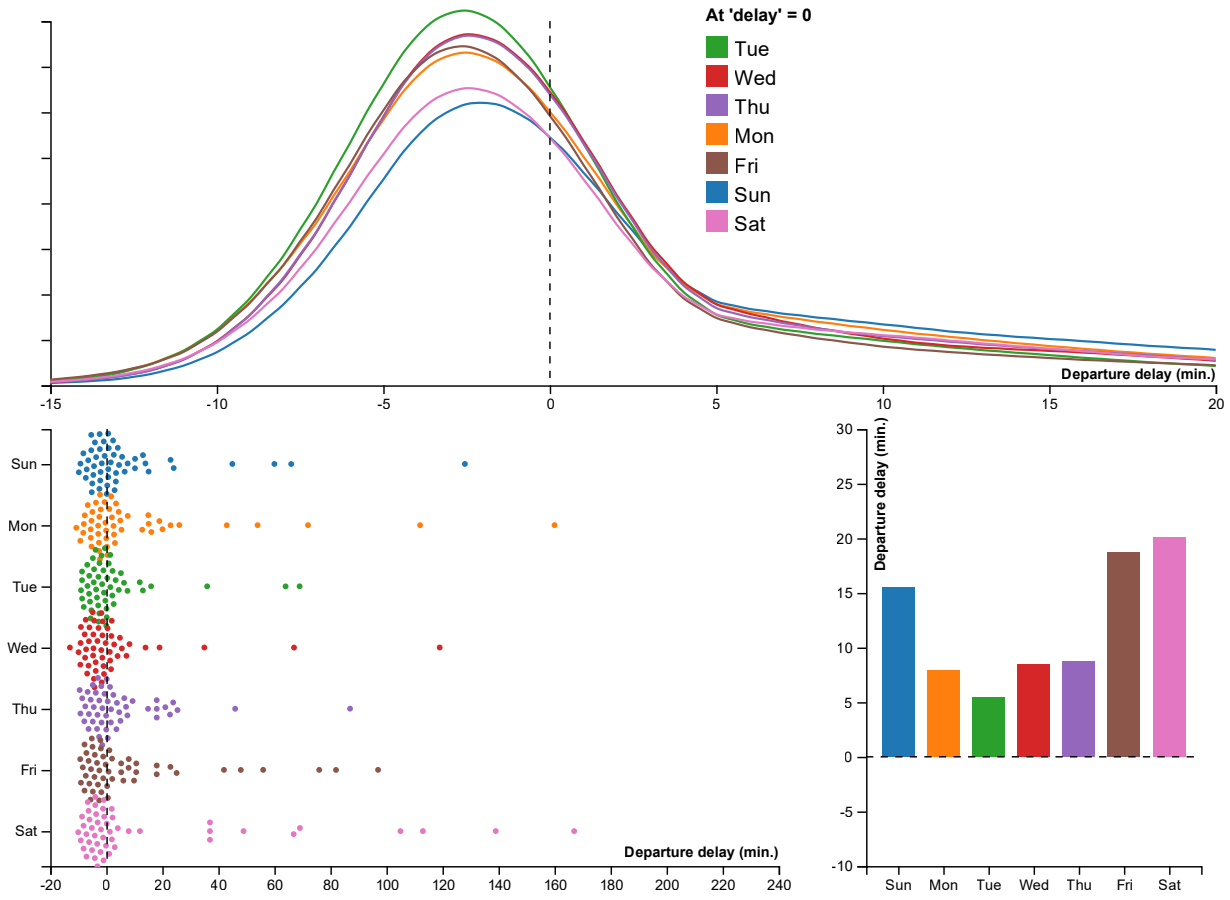


Figure 1.1: A dataset describing departure delay for a commercial airline is shown in three different ways. All visualizations show the delay depending on the day of the week. Depending on the reduction of the data, the visualization supports different tasks. Top, a kernel density function utilizing an Epanechnikov kernel aggregates relative densities of departure delay by day. Bottom-left, a beeswarm plot displays a subsampled set of the 71,486 flights total flights (50 per day). Bottom-right, a bar chart aggregates delay by taking the average delay per day. (Data from the United States Bureau of Transportation Statistics: shown is Delta Airlines for the month of December 2016.)

uses a visualization. As an example, Figure 1.1 demonstrates how different representations of flight delay data can change the information that a viewer can gather from a visualization. At top in Figure 1.1, flight delay data is aggregated using a kernel density estimator that demonstrates the distribution of delays per day. Subsampling (bottom-left) provides a discrete sense of distributions but may mis-represent outliers, while the bar chart (bottom-right) directly communicates a summary statistic (average) to compare delay. The method of summarizing data changes how a viewer can use a resulting visualization—for example,

the bar chart limits viewers to comparing statistical representations of the data, while the subsampled, beeswarm chart relies on viewers to generate their own representation of delays by day. A viewer can compare the distribution of delays by day with the line graph (top), while the bar chart (bottom-right) provides an average delay per day—the bar chart only allows the viewer to compare aggregate statistics together, while the line chart is more open-ended for the viewer to generate their own representation. These different types of affordances must be considered by the designer in choosing how to summarize data and design the resulting visualization.

The abstraction of data characteristics can also help to guide visualization appropriateness by cataloging features of the data that can affect the visual representation. Abstraction from domain-specific data characteristics, such as the number of positions in genomic population data (Chapter 6), generalizes the factors that affect effective design of a supporting visualization. These data characteristics can help to determine what visual encodings are appropriate for the attributes of the data [Bertin, 1983, Cleveland and McGill, 1984a], how to best encode multiple attributes of the data for the task [Shneiderman, 1996], and the design patterns utilized to display different types of data [Card et al., 1999, Craft and Cairns, 2005, Rind et al., 2016]. Most critically, this abstraction of data can allow visual solutions presented for a specific problem to be applicable to novel solutions in other domains, given a match in the abstractions. These abstractions can concern different characteristics of the data, such as the number of data elements to display (scale), the nature of the connectivity between the data elements (relationships), and the organization of the data elements (organization). By capturing these data characteristics, determinations about appropriate designs can be motivated.

Summary visualizations make design trade-offs in order to support different avenues of analysis. Figure 1.2 shows an example of summarizing the performance of a protein structural classifier. With the colors communicating the performance of the classifier (white and green are correct classifications, red and blue are incorrect), one can obtain an aggregate picture of classifier performance over this set of proteins (individual boxes), along with an idea of the error and variance in classification. The visualization also communicates the performance of the classifier on each protein—boxes that appear more red and blue

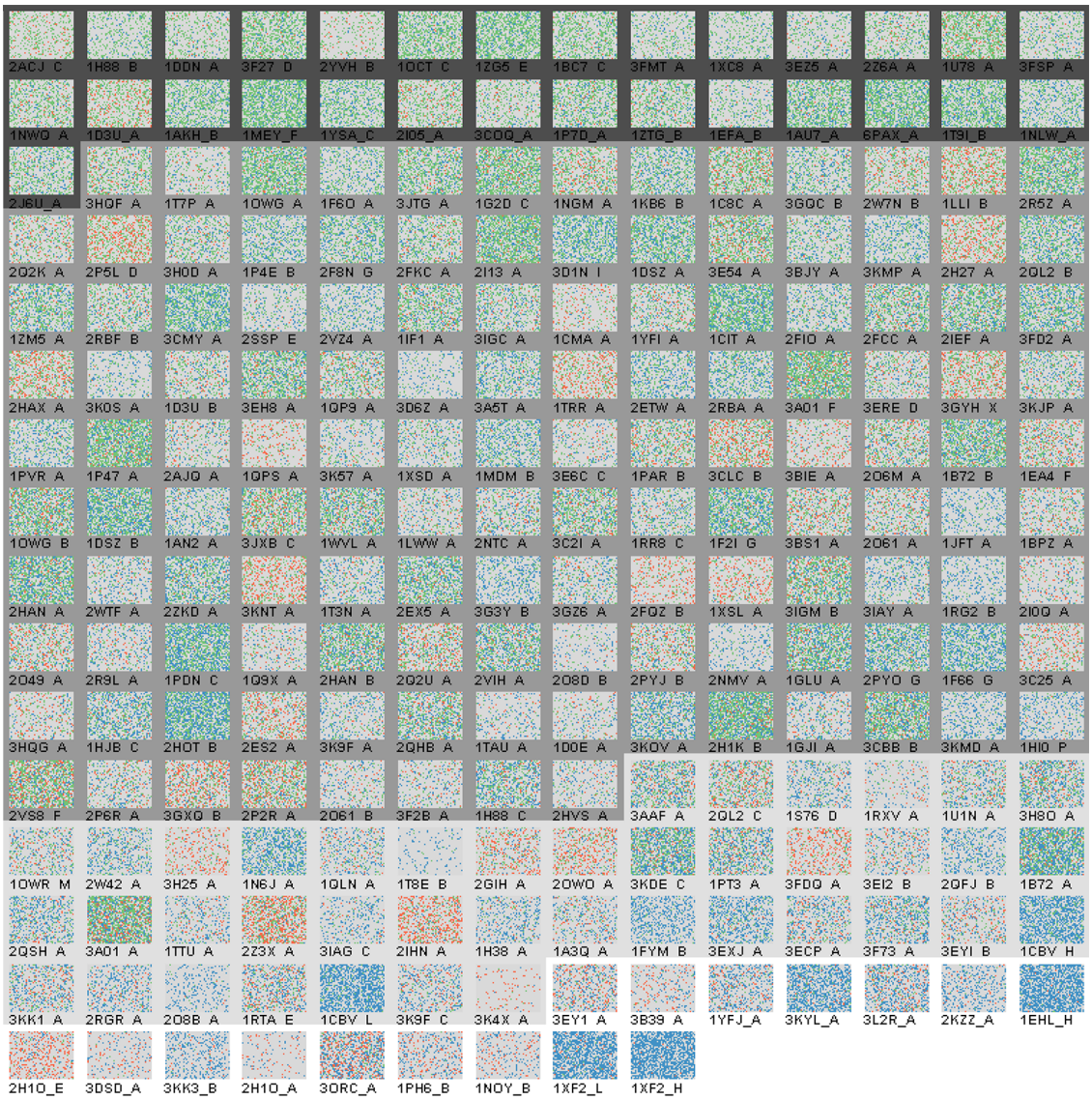


Figure 1.2: Sarikaya et al. [2014] presents a summary visualization of the results of a protein structural classifier, demonstrating the classification performance per protein. Color weaving [see Hagh-Shenas et al., 2007] is used to aggregate classifier performance for each protein (squares labeled by PDB identifier [Berman et al., 2000])—green identifies true positive decisions, blue for false positives, red for false negatives, and white for true negatives. Proteins that appear more green and white are classified correctly, while those that look red and blue are incorrectly classified.

generally have worse performance than those that appear green. Urness et al. [2003] introduced color weaving (adapted herein as woven blocks [Hagh-Shenas et al., 2007, Albers et al., 2014]) that emphasizes certain characteristics over others, and is appropriate for showing aggregate proportions of low numbers of categories. However, each protein

represented in this dataset are of different sizes, so the classifier makes a wide range of classifications per protein (from approx. 20–700). The use of woven blocks hides this potentially critical insight, and other visual strategies such as heatmaps may convey this through block size (see Chapter 5 for more detail). Understanding the trade-offs between design choices can help visualization designers and researchers make appropriate and effective design decisions for the task at hand.

Responsive to data scale and type, summary visualizations provide a high-level **overview** of a dataset, where overview provides the viewer with a visual representation that conveys the 'gist' of the data and can support targeted exploration of the dataset. Visualization, in contrast to explicit statistics, allows for the viewer to perform undirected search [Casner, 1991], supporting in-context exploration of the data and potential serendipitous discovery [Thudt et al., 2012a]. The design of these summary visualizations is guided by the requirements of the viewer (what do they want to know from the visualizations) and the necessary analysis tasks to satisfy those requirements, the characteristics of the dataset (e.g., how is the data organized), and how the data is minimized for display (summarization method). Therefore, it is my thesis that **principles for the design and implementation of summary visualization for collections of complex data elements are driven by the factors of the method of data summarization, complexity characteristics of the data elements, and the viewer's analysis tasks**. This dissertation describes both theoretical and practical underpinnings of these factors, and demonstrates how these principles can be applied in examples that, in concert, lead to effective visualization of complex data.

1.2 Addressing the Need for Guidelines

The issue of effective summary visualization is addressed in theoretical frameworks and practical implementation of visual analytic systems. At the highest level, a framework to codify how the different methods of summarizing data can affect the judgments of a visualized dataset can help to guide appropriate design decisions, also informed by the characteristics of the data. Understanding the trade-offs of using different summarization methods for collections of two-dimensional points (as commonly viewed with a scatterplot)

provides a concrete realization of visual summarization techniques. To realize novel visualization strategies, holistic design and evaluation of two instantiated systems are also presented to emphasize the utility of the theoretical frameworks. This method of generating generalized knowledge is part of the reductionist to holistic methodology of research [Correll et al., 2014], which evaluates both the theoretical underpinnings of design decisions and the cumulative effects of combining decisions in an instantiated visualization.

Such guidelines support the academic and practical considerations for visualization design. Developing guidelines can help researchers identify designs and techniques that can apply in cross-domain scenarios, and identify areas of the design space for which an effective design does not yet exist. From a pragmatic standpoint, guidelines can help visualization practitioners and designers to select appropriate designs, given a set of factors. Given the data and what information should be summarized from that data, guidelines help to inform designers about the possible space of designs while also highlighting the trade-offs between designs (e.g., a design may excel at identifying outliers, but at the expense of emphasizing distributions). These scenarios all indirectly help viewers of visualization by targeting the design of summary visualizations toward the communication goal—promoting accurate interpretations of a given dataset.

This dissertation makes the step of elevating visualization abstractions to be practical—demonstrating how theoretical frameworks can be made **actionable** for effective visualization design. The term 'actionable' refers to organizations that help to scaffold the iterative design process in visualization, abstracted to provide value for a wide range of analysis scenarios and data, and can potentially be ported to visualization authoring tools for practical use. Abstraction in visualization helps to both organize research so that knowledge can be collected to guide effective visualization and provide practitioners with an actionable methodology to make appropriate design decisions for their analysis goal [Craft and Cairns, 2005].

A critical component of effective abstraction can be motivated by the informed construction of *design patterns*, which can help to support reusable data visualizations across different data domains. Many "good" visualization systems are custom and bespoke, and ideally should be able to be reused for similar but different datasets both from the same data

source and from other data domains with similar data characteristics. By generating design patterns from instantiated designs and codifying their use and utility, the visualization community can build upon these patterns and select between options when designing and implementing novel visualizations. To support these design patterns, appropriate programmer abstraction can support these goals in practice, elevating theoretical abstraction to a practical level for immediate applicability.

1.3 Document Structure

This dissertation will describe these concepts in turn, starting with an overview of summary visualization and its abstractions along with relevant recent work (Chapter 2). I then address a theoretical framework for summary visualizations (Chapter 3), which is generated from instantiated designs in the literature, using a quantitative content analysis methodology. This theoretical framework organizes the methods of summarizing and reframing data against the information that a summary visualization conveys about the dataset. The factors of the method of summarization, the data type, the summary purpose, and the high-level affordances of the visualization are correlated to highlight design patterns and identify voids in the space of summary visualizations.

This study is followed up with a detailed design treatment of scatterplots (Chapter 4), where both the tasks and data characteristics are treated as factors that determine appropriate design strategies. In this work, the concept of a scatterplot is expanded to capture scatterplot-like designs, which all display data items as marks by two continuous dimensions, positioned in a two-dimensional space. With a survey of the literature and a card-sorting methodology, analysis tasks specific to scatterplots are selected, a collection of data characteristics that affect scatterplot design are identified, and a clustering of design decisions are collected and identified. To demonstrate the utility of this scatterplot-specific framework, we show two applications of our framework to organize the space of design decisions based on task and data, identify trends of task support with design strategies, and trends of task support as data characteristics (such as the number of items) change.

Two case studies of designing summary visualization are discussed from a summary

perspective, showing the design application to complex biological data (Chapter 5–6). Using the organization built up within this dissertation, I then discuss creating programming abstractions for authoring visualizations (Chapter 7), taking into account user intentions and different interaction paradigms for manipulating graphics. Lastly, the document discusses future work that could extend the work presented herein, particularly in the goal of summarizing data for the general public (Chapter 8).

This chapter brings together prior work relevant to this dissertation. The prose following serves to build up the context surrounding the problem of effective summary visualization proposed in the introduction, and motivate the need for the work presented within this dissertation by highlighting the gaps in the existing literature. Related work specific to the individual chapters will also be presented in their respective chapters.

2.1 Summarizing Data in Visualization

As datasets continue to grow in size and complexity, the design decisions made to support summary visualizations need to scale to these challenges. What exactly needs to be summarized depends on the characteristics of the data the viewer wants to observe and how a viewer expects to interact with the data. In order to evaluate the effectiveness of a summary visualization, it is helpful to understand how the concept of **overview** is used in the visualization literature. As defined by Spence [2007], overview is "a *qualitative awareness* of *one aspect* of some data, preferably acquired *rapidly* and, even better, pre-attentively: that is, without cognitive effort." This definition is quite focused on obtaining a single factlet over an 'overall sense' or 'gist' of a dataset. In colloquial visualization parlance, an effective overview imparts an awareness of the data and its trends. Due to this ambiguous definition, exactly *what* constitutes an effective overview or summary of a dataset is quite dependent on the domain and specific characteristics of the visualization under test. Hornbæk and Hertzum [2011] presents a part-synthesis, part-survey methodology to understand the concepts surrounding overview, and how visualization research supports this summarizing notion. The chief difficulty in this area is providing generalizable guidance that works for a variety of analysis scenarios, as well as for a diversity of data domains. I use **summary** in this document (as opposed to 'overview') to consider and organize design considerations in a more concrete manner.

A summary visualization seeks to provide high-level information about a dataset, but not necessarily a complete overview (e.g., communicating "all" high-level information).

As summarization in visualization is considered to involve an explicit decision to reduce data for display, it is critical to understand how the method of reducing data correlations with the information communicated by a resulting visualization. However, when designed effectively, such visualizations can elicit insights that even the designers did not anticipate. Thudt et al. [2012a] demonstrate in their paper that when the overview is designed to place similar but potentially discongruous items together, viewers modified their exploration process, eliciting new hypotheses. Kosara [2016] show that the utility of visualizations change based on when they are explicitly used for presentation—different designs work better in different scenarios. While Kosara identifies this distinction, I seek within this thesis to identify how the differences manifest themselves, and provide guidance for effective design, with respect to the characteristics of the data, the type of data summary, and the analysis tasks of the viewer. When considering these factors, it is critical to identify the ramifications of poorly-targeted summaries, which allows for the critique and evaluation of summary design. These ramifications include issues with object overdraw, the design and methods of data summarization, as well as methods of reusability and evaluation.

The issue of occlusion or **overdraw** occurs when the visual mark representing individual data items completely overlap one another. A common example of overdraw occurs in scatterplots when the number of items overwhelms the available screen-space for the plot, and the marks occlude one another. In this scenario, it is difficult to make judgments of relative numerosity—how many points are in this region against that region? For multiple series of points, what are the proportions in the plot when points are drawn over each other? Overcoming the technical issues of overdraw, as explored by Fekete and Plaisant [2002] in their seminal work, allows summary visualization to scale to very large datasets, with the trade-off of sacrificing fidelity in some types of judgments to achieve these scales. From the human-centered standpoint, Cui et al. [2006] tackles the understandability issues that overdraw exacerbates, including ambiguity in the analysis task or goal, especially in how designs can greatly limit how a summary can be used. Thus, for effective summary design, overdraw must be minimized as it inhibits visualization utility.

Summarizing data before visualization can help to offset the problem of overdraw. In this thesis, I explore how different methods of data summarization can affect the affordances

of a derived visual summary. While a detailed discussion of these different methods occurs later in this document (Chapter 3), a few classic methods are presented here. Cleveland and McGill [1984b] describes several different design strategies to simplify the design of a scatterplot, including regular lattice binning with glyphs, adding smoothings (continuous average trendlines), and representing multiple series with icons, shapes, and color—see also Cleveland, 1985, chap. 3. Card and Mackinlay [1997] has an early collection of data reduction methods, but tends to mix high-level categorizations with low-level methods, such as identifying filtering, sorting, slider selection, and mutli-dimensional scaling. Ellis and Dix [2007] collects and derives a taxonomy of clutter-reduction techniques, including methods such as clustering, sampling, and topological distortion. While such a taxonomy may be too general for practical use, the authors raise important issues in data summarization for visualization, including how filtering, sampling, and clustering limits potential insights of the data.

For more specific methods, Bertini and Santucci [2006] describe the use of subsampling to understand patterns in the data by stochastically removing data to decrease the prevalence of overdraw. Elmqvist and Fekete [2010] describes how summaries can take advantage of hierarchies in the data to group and otherwise target summary design. The authors describe a two-pronged approach, where the data can be clustered based on hierarchical ordering, and the visual design can also utilize a hierarchical design. These methods help to concretize our four methods of data summarization—aggregation, filtering, subsampling, and projection—which are discussed in Chapter 3.

Broad guidelines have been introduced and have worked their way into the lexicon of data visualization. The oft-cited "Information Visualization Seeking Mantra" by Shneiderman [1996] states that visualizations should be designed to provide "overview first, zoom and filter, then details-on-demand." The idea behind the mantra is to design the visualization such that a viewer obtains an overview, has the ability to drill into the phenomenon they are interested in, and then can use the visualization to develop or confirm a hypothesis about the data. With great attention from the visualization community, Craft and Cairns [2005] provide a counter-point to this mantra, where the un-validated mantra has been used to justify a very large number of design decisions. Despite the use of the

mantra, Craft and Cairns discuss the need for actionable guidelines that are also high-level and domain-independent, particularly in prompting predictive summary and interaction design paradigms for the benefit of the viewer.

The need for guidelines for design in the research literature have become clear, and many guidelines have been developed for particular analysis scenarios or data domains. These guidelines underscore the need for guidelines for summary visualizations, much as these works have influenced design of their respective domains. Leung and Apperley [1993] introduced a framework for evaluating visualizations that summarize datasets, proposing a cost function that evaluates the accuracy of insights generated by the overview provided by the visualization. van Wijk [2005] extends the framework to practical problems, and the cost guideline has been extended by subsequent work [*cf.* Kindlmann and Scheidegger, 2014] to compare the information conveyed from different design decisions. More specific design-centric guidelines for summary visualizations have also been proposed, including designing for pre-attentive pop-out [Haroz and Whitney, 2012] (to support highlighting discrepancies and proportions) and designing for rapid re-ordering of the display [Slingsby et al., 2009], where re-organization can help answer a wide range of viewer queries on the data.

2.2 Taxonomies and Organization of the Visualization Design Space

In the history of data visualization research, there have been many organizations of the conceptual design space of visualization. Perhaps the original organization of the space rests with Bertin, introducing the concept of visual variables and their use in a built visualization [1983]. Bertin also introduced the dichotomy present in visualizations—the *purpose* of a visualization can be for presentation (demonstrating previously understood or curated insights) or exploratory (addressing new or generative questions of the data). The question of how to best support novel exploration of the data was explored in the early '90s, with Springmeyer et al. [1992] describing the scientific data analysis process: investigation

(involving interaction, application, and maneuvering within a visualization), generation of insight (targeted exploration and confirmation through details), and interaction with the overview (including the operations of examine, orient, query, compare, and classify). Mullins and Treu [1993] extend the scientific *sensemaking* process to capture the interplay between the analysis, synthesis, and assessment of insights from visual data analysis. This framework highlights the pivoting of the field to support exploratory and confirmatory visualizations for data analysis.

To better capture how viewers use visualizations, task-driven methodologies strive to capture how a visualization is use and should be designed to support the intentions of the viewer. Roth and Mattis [1990] is an early example of capturing the tasks of the viewer and discusses the presentation functions (nominally, overview tasks) such as lookup, comparison within and among relations, the distribution of relations, as well as the ordering and correlation of attributes, many of which are difficult to obtain through statistical or numerical means. Casner [1991] extends this to perceptual operators such as (non)targeted search, verification, and lookup—separating viewer-known from unknown patterns in the data. Zhou and Feiner [1998] extends the presentation versus exploration dichotomy, adding iterative elaboration and verification as a core task, and proposing that summarization and searching comprise different intents on behalf of the viewer.

Klein et al. [2006] and Amar et al. [2005] propose a viewer intent-driven taxonomy, where the analysis intent of the viewer helps to categorize what is done with a visualization. Klein et al. proposes a data frame theory of sensemaking, where the viewer goes through stages of recognizing, elaboration, questioning, and reframing the visualization to support a new data frame. Amar et al., in contrast, collect and abstract a collection of viewer tasks such as "retrieve value", "determine range", and "find anomalies"—while this list does not claim to be comprehensible, the work sets off a barrage of follow-up work that seek to evaluate how visualizations are or are not appropriate, given the tasks that need to be supported for the design to be effective. Andrienko and Andrienko [2006] introduce a very detailed collection of methods for obtaining information from complex data displays, concentrating on directed and undirected search of data items and their relations.

These task taxonomies provide a representative list of actions and intents that a viewer

may want to perform with a visualization. Depending on how the visualization is used by the viewer, the performance of accomplishing these tasks can help to rank design decisions in their effectiveness for the given task. It is well-established that no one set of design decisions supports the full gamut of analysis tasks—design decisions have trade-offs—a set of design decisions may support some tasks at the expense of others. Therefore, the designs of many visualizations are targeted—supporting a limited set of tasks for a particular set of analyses. Munzner [2009] describes a nested model for visualization design with this in mind, emphasizing that the goal of the visualization must match what the viewers expect, from the workflow down to the visual variables used in the visualization. The core of the evaluation centers on capturing the tasks of the visualization at a high-level (e.g., "identify the genomic positions that have mutation correlation") and at a low-level (e.g., "identify the frequently mutating genomic positions"). While the distinction between high- and low-level tasks can be ambiguous in practice and use, high-level tasks tend to be a composition of many low-level tasks—the "ends" to the "means."

High-level task organization

Recent task methodologies have approached the abstraction of analysis tasks from a holistic perspective. With this perspective, these organizations can help to standardize the discussion of how tasks affect visualization design. Heer and Shneiderman [2012] explore how analysis tasks can be supported through interaction with the visualization, mostly by shifting the frame of reference (much like Klein et al. [2006]) or by re-organizing the data to arrange the data to be suit a given task. To bring organization to the taxonomies of tasks themselves, both Brehmer and Munzner [2013] and Schulz et al. [2013] present a hierarchical ordering to capture a wide range of high- and low-level tasks, though they utilize slightly different strategies. Brehmer and Munzner collect a wide range of viewer tasks from the literature and organize the tasks by *why*, *how*, and *what*—which themselves are further broken down. The *why* category abstracts the intent of the viewer: how the visualization is consumed (present, discover, enjoy; an extension of Bertin [1983]), the method of search (similar to Casner [1991]), and query pattern (similar to Andrienko and Andrienko [2006]). The *how* derives directly from many low-level tasks taxonomies

discussed earlier, and provide the means to support the viewer in their analysis intent. To connect multiple tasks together, the *what* can capture the information generated from previous tasks to inform future tasks.

Concurrently published, Schulz et al. [2013] use a different yet related methodology, expanding the "ends" of Brehmer and Munzner [2013] to capture the *goal*, "means" as high-level *means* (is data elaborated, re-organized, or put in context for comparison?), *characteristics* (e.g., data values or trends), *target* (attribute and structural relations), and *cardinality* (the scope of the data instances: single, multiple, or all; also discussed by Andrienko et al. [2003]). With Schulz et al.'s framework, tasks are identified as "points" in the five-dimensional design space—bounding the space of all tasks, and providing a common abstraction for viewer tasks performed with all visualization. This organization is attractive in its multi-layered description of all tasks, and the organization allows for the abstraction and relation of any analysis-specific task into this space. This common organization helps to identify similarities between seemingly disparate analysis tasks, helping to realize common abstract tasks. Throughout the next two chapters (Chapters 3–4), I will use Schulz et al.'s approach to guide our capture of viewer analysis tasks.

Most relevant to the arguments in this dissertation is how these organizations of analysis tasks (and the tasks themselves) drive the design of visualization. This dissertation continues this important work, and builds upon this burgeoning area. While Munzner [2014] collects a task-driven methodology for making effective visualizations in her textbook, specific research is described here. Ji Soo Yi et al. [2007] discuss tasks that viewers perform when interacting with a visualization to elaborate or summarize data, and suggest design elements that support those viewer intents. Pike et al. [2009] take these intents a step further by connecting the viewer's goals and tasks with the interaction affordances designed for a visualization. The authors emphasize that, while visualization techniques are ever-changing, dynamically capturing the intents of the viewer can allow for semi-automatic re-organization of the data and design decisions to support the changing goals of the viewer. This future work represents one of many potential benefits from connecting analysis tasks with design decisions. In the spirit of the dissertation, Rind et al. [2016] explore how support for viewer tasks drive visualization design. Rind et al. discriminate

the dual-role of tasks to drive design decisions, and to evaluate the effectiveness of a visualization. In this dissertation, tasks are used as the means to evaluate the appropriateness of design decisions in a given visualization, whether for the generalized case of summary visualization, a specific visualization such as a scatterplot, or particular instantiations such as verification of a predictive protein classifier.

2.3 Abstractions for the Visualization Designer

Concept abstraction is widely used in the data visualization literature. Abstraction helps to take specialized model tasks and generate a generalized representation that can be applied to other analysis domains and for different structures of data. This section references those abstractions and concept organizations that are helpful for framing the visualization design process. Here, concepts that are used throughout the dissertation are discussed: the sensemaking loop, explicit versus implicit statistics, and how these organizations can be designed to be actionable.

The sensemaking loop describes how viewers use visualizations, and how the viewer should be brought into consideration when evaluating the effectiveness of a visualization. This is of particular importance to the iterative design process of designing summary visualizations for the two use cases presented in this dissertation (Chapters 5–6). The comparison of explicit against implicit statistics highlights issues of depending on statistics to convey high-level information about a dataset, particularly in exploratory contexts (as discussed throughout the dissertation). In closing, I highlight actionable organizations that have been created in the visualization literature that inform both researchers and practitioners to create and design effective summary visualizations, such organizations that I seek to contribute with this document.

2.3.1 The Sensemaking Loop

Sensemaking is the process by which a viewer interacts with a visual representation (or more generally, the environment), and builds and revises their internal model of the represented data (and generally, the world). The hope of designers and researchers is that

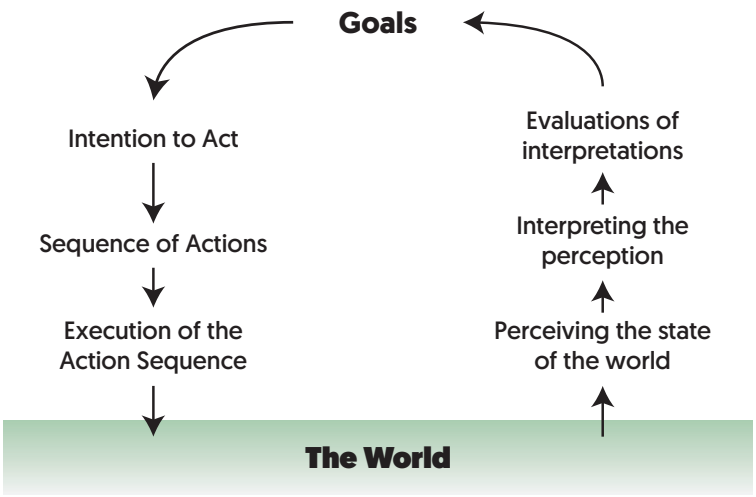


Figure 2.1: A diagram of Norman's Action Model [1988], describing how an individual interprets and builds their knowledge about the world about them. In terms of a viewer interacting with a visualization, the left side captures how the viewer's mental model prepares them to interact with the visualization (from experience with the data and/or visualization mechanics). The right side describes the process by which the viewer interprets the information, updates their mental model, and potentially generates new goals to begin the action loop anew.

by understanding the sensemaking loop, better representations of data can be made to support viewers in their analysis and to quickly confirm hypotheses or disprove questions of the visualized data. Much of the work in sensemaking derives their core model from Norman's action model [1988], which demonstrates how a human (in our case, a viewer) approaches the world (in our case, data externalization or visualization), interacts with it, reconciles their anticipation of (external) results with their internal model, then updates their internal model by rationalizing the unanticipated phenomenon. Figure 2.1 shows a diagram of this action model.

This action model has been adapted by many authors to explain how viewers interact with data visualization, perhaps most notably by Pirolli and Card in their 2005 paper that augments Norman's action model for use in "analyst technology" (which is used to externalize data; essentially, the role of visualization). They detail an explicit process of iteratively seeking information by collecting data from sources[1] and extracting the

[1]While obtaining relevant datasets for a given analysis scenario is a critical component of effective analysis, it is out of scope for this dissertation (it is assumed that all relevant data has already been gathered).

relevant data (*foraging loop*), then iteratively developing a mental model by schematizing and reconciliation with existing experiences, and finally evaluating the data based on generated hypotheses (*sensemaking loop*). Within in the sensemaking loop, Pirolli and Card generalize the process as Information → Scheme → Insight → Product. Though the language is obtuse, the key theme is the *insight* that leads to a productive outcome (*product*). The concept of *insight* has been co-opted in the information visualization literature to identify "observations" or "breakthroughs" that leads to "discovery"—explained as "seeing something that previously passed unnoticed or seeing something familiar in a new light" [Saraiya et al., 2005, p. 443]. It may follow that the purpose of visualization is to support and maximize the potential to generate insights on behalf of the viewer [*cf.* introductions of Card et al., 1999, Spence, 2007].

If maximizing the potential for insights is the goal, it is imperative that the initial state of the viewer's mental model must be understood to design an effective visualization. Saraiya et al. [2006] note that insights from exploratory visualizations are constructed based on both previous experiences with visualization and the data domain. For this reason, many specialized visualizations are evaluated by domain experts who have expertise in the domain [Pirolli and Card, 2005]. Liu and Stasko [2010] take a great step to bring together mental models (in general), visual reasoning, and interaction in visualization together to make these concepts actionable for the designer. With regards to mental models, Liu and Stasko notes that they are a "functional analogue representation to an external interactive visualization system" (p. 1001) that captures the structural and behavioral properties of the visualization, can preserve schematic and semantic information about the data, and that the mental model can be used to reason about the data. This speaks towards the design of effective visualization—it must be understandable to the viewer (they must have some familiarity with the data or visualization), and it should allow the viewer to manipulate the view to obtain an anchoring point [Liu and Stasko, 2010, see Sec. 5.2.1].

Using this theory of the mental model, visualizations should support viewers' mental models, whether they come from domain expertise or individual differences [Ziemkiewicz et al., 2012]. One method for doing this is supporting switching from familiar representations (which may not effectively support a given task) to an alternate representation (that

has greater effectiveness) [see the Data/Frame theory, Klein et al., 2006]. This supports designing for existing models of interacting with data, using accessible and familiar visualizations such as scatterplots. Associated interaction mechanisms help to change the reference frame, focusing the viewer's analysis and generating different representations of the data. From these new representations (whether using a different visual representation or summarizing the data by some method), viewers can approach their data from a new angle and continue their foraging [Liu and Stasko, 2010]. Yi et al. [2008] discuss how to organize interaction in a visualization to support the creation of insights. While not mutually exclusive to one another, Yi et al. identify themes that interaction supports: providing overview, adjust, detect pattern, and matching to mental model. This dissertation directly focuses on supporting these viewer interactions (whether supported solely by visual design or supplanted visualization interaction)—by understanding how data summarization can affect what types of features viewers can see in the data, and how tasks, data, and designs are related to one another.

Of relevance to the evolution of the work presented herein, Shneiderman and Plaisant [2015] discuss the need to sharpen analytical focus, particularly when dealing with "big data." They discuss the issues in providing relevant summary visualization when dealing with large amounts of repetitive data—there must be a way for the viewer to identify and focus their attention on a subset of the full dataset. This focus could be on an exemplar set of data or performing cohort analysis between multiple organizations within the data. Understanding how best to support an analyst's exploration through a dataset would help to choose the necessary interaction and design strategies for effective visualization.

2.3.2 Explicit Versus Implicit Statistics

Summarizing data necessarily entails throwing away data. **Explicit** statistical marks can capture high-level summary information about a dataset. On the extreme end, Potter et al. [2010] demonstrates visualizing many different explicit statistics of many sets of one-dimensional distributions, including mean, skew, one and two standard deviations, and kurtosis. Capturing these explicit statistical metrics can help viewers accomplish their analysis goals, but also have the potential to leave exclude necessary information.

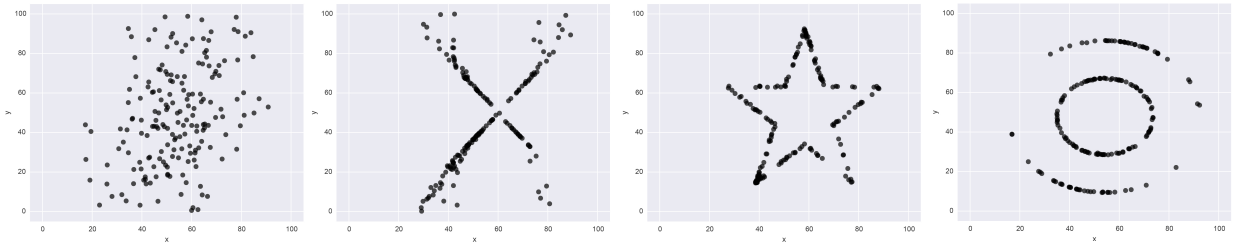


Figure 2.2: Multiple sets of data have noticeably different distributions, but have identical summary statistics (for this group: mean: $[54.26, 47.83]$, standard deviation: $[16.76, 26.93]$, correlation: -0.06). Only by visualizing the data can differences between datasets be seen. Data and graphs from Matejka and Fitzmaurice [2017].

By nature of summarization, individual item fidelity must be sacrificed—as fidelity is removed through data summarization, it may be impossible to recover particular details. If the viewer's goals are known *a priori*, explicit statistical summarization is a concise method of communicating distributional characteristics.

However, Anscombe's quartet [1973] (and its derivatives, such as Matejka and Fitzmaurice [2017]) illustrate pitfalls of reliance on these explicit statistical measures. The set of graphs in Figure 2.2 have very similar statistics, but represent very different distributions of data. A benefit of visualizing such data in a relevant subspace is the ability to see and compare distributions, especially when the relevant characteristics are not known to the viewer *a priori*. To build a mental model of the data, visualizing *implicit statistics* (in contrast to explicitly-marked statistics) may be desirable. The implicit nature is derived from Gestalt grouping [Wertheimer, 1923] from appropriate design choices for a given analysis task.

Implicit statistics can take advantage of the viewer's preattentive process, whereby the human visual system identifies and prioritizes particular elements in a display [Ware, 2012]. By taking advantage of these preattentive mechanisms, trends and anomalies in the data can be emphasized in the visual display of large datasets. To understand how these mechanisms work, there has been a multitude of empirical and holistic work in both perceptual psychology and visualization. Ariely [2001] demonstrates some of these preattentive mechanisms and potential confounds. In particular, Ariely highlights with three experiments how a viewer creates a representation of a complex visual set of data—with short stimulus times, viewers tend not to recall properties of individual items but

instead create a representation of the set. Gleicher et al. [2013] further explored how viewers create these representations of sets with many items, and found that these mechanisms are not limited by the number of objects. In fact, they found that more objects within a set tended to strengthen judgments about distributional comparisons between multiple sets, which emphasizes the scalable power of the visual system.

In this same vein, Halberda et al. [2006] also show that the number of items in a display has less relevance to internal representation than the number of "selectable" groups, finding that the visual system can hold around three set representations in short-term memory. For the types of encodings that create groupings, Healey and Enns [2012] provide a comprehensive overview of visualization-targeted mechanisms. They survey many types of features that preattentive processes quickly identify, and highlight strategies for taking advantage of these properties. Of potential relevance to visualization, Healey and Enns show that ensemble (en)coding strategies and feature hierarchy can be engineered by designing combinations of preattentive features together. By doing so, designers can design to emphasize particular visual features of data at scale.

From a holistic perspective, Javed et al. [2010] identifies preattentive features between different representations of line graphs that yield different viewer interpretations of the data. They empirically evaluate different ensemble strategies (established visualization types such as line graphs, small multiples, braided graphs, and horizon graphs) based on different analysis tasks, and establish a set of affordance trade-offs between the designs. Similarly, Haroz and Whitney [2012] highlights the importance of ordering items to emphasize trends in group membership and highlight anomalies in the data. They note that even though the analysis task may not be known *a priori*, grouping can greatly help identifying an unknown target by emphasizing discontinuity.

Of relevance to creating summary visualizations, both Correll et al. [2012] and Albers et al. [2014] explore the idea of *visual aggregation* to communicate high-level information about time-series data. Visual aggregation comprise a set of techniques that coax viewer's visual system into creating representations of data distributions and trends. Correll et al. [2012] show that communicating time-series data using color weaving [Urness et al., 2003] can impart a more nuanced internal representation of the data, including identifying high-

variance time periods from others. This work is extended by Albers et al. [2014] to capture a range of tasks done with line charts, which include identifying the extremes of the data, aggregating values over ranges, and identifying high- and low-variance regions of values. The authors empirically evaluate different encodings (weaving, color blending, event striping) and find that different encodings support different types of tasks with varying accuracy. In general, position encoding that use linear structure (line charts, box plots) tended to support higher-accuracy viewer judgments, while aggregate color encodings supported viewers in making accurate representations of average value over a range. With this evidence of visual aggregation, Correll and Gleicher [2015] argue that implicit statistics can equally support viewer judgments and representations of data, particularly in situations where the viewer's overview goals are not known *a priori*. In a summative article, Szafir et al. [2016] describe different ensemble encoding strategies to support viewer judgments of implicit summary statistics.

The power of implicit statistics is used throughout this dissertation to communicate high-level information about a given dataset. Chapter 3 discusses how different methods of reducing data can affect the types of high-level information communicated by a resulting visualization, while Chapter 4 describes how both implicit and explicit statistics can be shown in scatterplot-like designs.

2.3.3 Actionable Organizations

This dissertation focuses on creating actionable organizations that can be utilized by both researchers and practitioners of visualization. These organizations can help practitioners understand the factors that affect trade-offs between different design strategies, and help researchers organize and position their novel techniques in relation to previous work.

Examples of organizations that have influenced the use of abstractions in data visualization that appear within this document visualization include Munzner's nested model paper [2009], which separates targeting design and visual techniques and proposes a methodology to evaluate the efficacy of different levels of a visualization independently. Though considered a "theory" paper, these ideas are utilized in the design process for many application papers (e.g., Chapter 5). Elmqvist and Fekete [2010] looks at how aggregation

can be exploited in the data and the visualization design to simplify viewer interaction with large, complex data. They provide guidelines for designing overviews of the data to preserve particular characteristics of the underlying dataset—and one of their guidelines is to design for "visual summary", though no guidance is given for what makes an effective summary. One of the goals of this dissertation is to extend these organizations to make them actionable for summary visualization design.

The task taxonomies of Brehmer and Munzner [2013] and Schulz et al. [2013] help to collect and abstract the analysis tasks that viewers perform with visualization. These two collections (along with other collections such as Andrienko and Andrienko [2006] and Roth [2013b]) aggregate information from a whole host of earlier, specialized task taxonomies, covering a wide swath of tasks (see §3.2 for more detail). If a sufficient set of representative tasks are evaluated against design decisions and other relevant factors, tasks can become a driving force for supporting effective visualization design. This core idea is used to motivate each of the following chapters in this dissertation.

3 DESIGN FACTORS FOR SUMMARY VISUALIZATION

Visual analytics systems help users navigate large and complex datasets. These datasets often have too much data or too many dimensions to display in one view, requiring designers to engineer systems to first *summarize* available data and then visualize the results. The resulting *summary visualizations* help orient analysts by describing high-level information about the dataset, guiding analyses of particular features, and providing a means for navigating to important subsets of data. Designers and researchers have numerous techniques and design choices for constructing these summary visualizations, but little systematic guidance for reasoning about the trade-offs of different design decisions and their impact on the resulting analyses.

In this chapter, we survey summarization in visual analytics, evaluating the relationships between use, analytic affordances, and data summarization methods. We aim to understand, recognize, and characterize limitations in current design choices for effectively summarizing data for visual exploration, focusing on factors such as analysis tasks, data types, and data characteristics. We provide an abstraction of existing summarization methods to support different analyses, and use this survey to propose an initial design space for data summarization for visual analytics. Understanding common links between tasks, data, and techniques used to summarize data for analysis will help guide new tools and opportunities for innovation in visual analytics systems, and for summary visualizations in general.

Summarization in visual analytics serves two primary purposes: to compress the dataset to fit in the available screen-space and to reduce visual complexity to make visualizations easier to interpret. Examples of summary visualizations include histograms that aggregate data across a selected dimension, dimensionally-reduced scatterplots that project high-dimensional data into a lower-dimensional space (see §4), and actor-network diagrams that summarize relationships between entities captured in a text corpora. Summarization is an essential component in most visual analytics tools—we found that more than half of papers surveyed contained a summary visualization. The ubiquity of summarization and its impact on data analytics tools means that designers need a better understanding of the factors that lead to effective summarization to guide the design of effective visualization

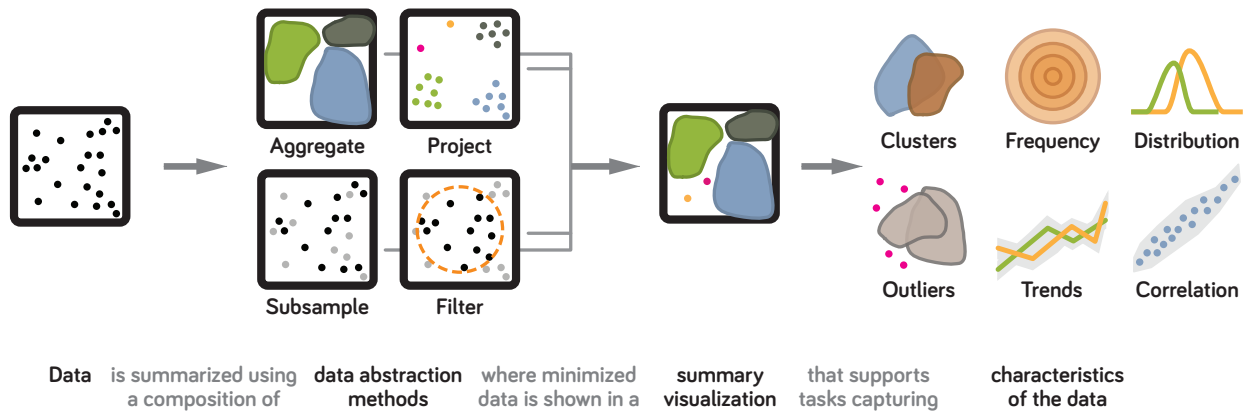


Figure 3.1: A schematic of a generalized process for visual analytics with data summarization. A dataset (left) is reduced using data summarization techniques (center), comprised of four basic methods (aggregate, project, subsample, filter), and is presented visually to support judgments of high-level data characteristics (right). Both the summarization and visual presentation are factors that influence the efficacy of summary visualization to enable viewers to make high-level judgments.

tools. We construct a preliminary design space of summarization for visual analysis (Figure 5.1) that allows designers to reason how factors are involved in affecting the resulting utility of a summary visualization.

We systematically survey the visualization literature to observe relationships between factors in this design space using quantitative content analysis (QCA) [see Riffe et al., 1998] to analyze summarization in IEEE VIS and EuroVis papers from 2009 to 2015. Our approach quantifies the relationship between analysis tasks, properties of the data, and summarization choices to identify design themes in summary visualizations. Our goal herein is to confirm that the list of summarization methods is sufficient to capture all methods of summarization and also to understand how the method of summarization affects the affordances of a resulting visualization. We use our design space analysis to identify common *themes* observed in summary visualization design. These themes indicate common patterns in how designers *use* summarization to guide analysis, a preference for task *specificity*, the existence of common *design patterns*, and a bias towards certain design choices for specific types of *data*. These themes highlight key considerations for data summarization, identifying *challenges* in current practices and *opportunities* for new and innovative thinking around summarization in visual analytics.

Contributions: We conduct a systematic survey using QCA to characterize summarization in visual analytics as a function of purpose, data summarization method, analysis task, and data type. In doing so, we make the following contributions:

- A taxonomy of data summarization techniques used in summary visualization (§3.1–3.2),
- A formal survey and analysis of summary design in exploratory tools (§3.3), and
- A description of challenges in summary design (Table 3.2) and potential opportunities for innovation (§3.3–3.4) grounded in existing practice.

This chapter provides a foundation for systematically reasoning about summarization in current and future tools and identifies gaps in our general understanding of summarization in visualization design.

3.1 Background

Visual analytics systems often provide summaries that analysts use to navigate, sift, and winnow through data to create a concise and focused representation of the underlying dataset [Shneiderman, 1996]. A summary within a visual analytics tool communicates properties of a dataset by explicitly using fewer marks than there are datapoints. For example, a scatterplot with points aggregated using KDE could constitute a summary, whereas 'zoomed-out' representations of a dataset, such as a standard scatterplot or parallel coordinates plot with many thousands of elements but no data minimization would not qualify as a summary. While individual points may be difficult to distinguish in such zoomed-out representations due to factors such as overdraw (see Fekete and Plaisant [2002] for technical issues, Cui et al. [2006] for understandability issues), such visualizations do not procedurally summarize data. Instead, we focus on methods that explictly summarize data (a strategy most overdraw solutions employ) and analyze how the ways data is summarized and presented affect high-level judgments of the visualized data.

In this work, we consider a **summary visualization** the result of an explicit set of summarization decisions made by the designer, together with the reduced data and the

visual representation. Throughout this dissertation, the role of summaries is to convey a "gist" about global and high-level properties of a dataset, as discussed in Section 2.1. Design of these summaries should ideally support the needs of the analyst or audience, but the data type, the method of reducing data, and the anticipated use can all affect the resulting design. We draw on these prior characterizations of overview and our own observations to organize the design space of how summaries are used in visualization to generated a grounded codebook for QCA (§3.2.2). From this literature, we propose a design space characterized by *purpose*, *summarization method*, and *task* that guides the design of effective summarization, and discuss these organizations in detail. By basing codes on previous work, we can use these organizations in our effort to observe relationships between these factors for effective summary design.

3.1.1 Data

The type of *data* affects how data is summarized and what global, high-level features an analyst can extract from a summary. Taking note of previous approaches, we can observe how different techniques can affect a resulting visualization based on the hierarchical organization for data, data with multiple dimensions, and dealing with large amounts of data. Hierarchical data can be summarized by visualizing data at different levels within that hierarchy. Elmqvist and Fekete [2010] survey how aggregation techniques, in particular, can reduce the amount and complexity of visualized data. They demonstrate how hierarchical aggregation can be applied to conventional visualization types, even for non-hierarchical data types, and also provide guidelines for effective navigation within hierarchically aggregated visualizations. Elmqvist and Fekete provide a guideline of *visual summary* (G2) that "visual aggregates should convey information about the underlying data." We consider the effect that aggregation can have on a resulting summary, and also consider how a summary is affected by a broader set of transformation and organizations of non-hierarchical data.

While other types of data have unique challenges for summarization, a common theme in summarization is dealing with high-dimensional and spatial data. Both Keim and Kriegel [1996] and Keim [2002] have explored visualization techniques for exploring databases,

where visual interfaces summarize datapoints and their attributes using overview first, enabling the viewer to explore large amounts of data. Kehrer and Hauser [2013] have surveyed the high-level design and intents of visual analytics overview approaches, exclusively for multifaceted scientific data. They identify many techniques in their survey that lead to summaries for particular types of data, but do not directly draw conclusions about the affordances of different techniques and the cross-applicability of summary designs for different data domains. Leung and Apperley [1993] provide a framework for evaluating visualizations where there is too much data to display each datapoint clearly. This framework helps designers evaluate visual and computational representations of summaries based on their *effectiveness*, *expressiveness*, and *efficiency*; however, it provides no guidance as to how representations might be designed with these qualities in mind.

3.1.2 Purpose

The *purpose* of a visualization describes its intended use. We anticipate that the intended purpose of a summary directly informs effective design. Bertin [1983] presents purpose as a dichotomy: the visualization either communicates previously understood information (presentation-oriented visualization) or supports information processing to address new questions (exploratory visualization). Schulz et al. [2013] refines this division to consider the goals of an analysis: *exploratory*, *confirmatory*, and *presentation*.

We hypothesize that summaries for presentation emphasize specific data characteristics more often than exploratory summaries, and that the intended purpose of a summary (exploratory, confirmatory, and presentation) can inform effective data summarization. This division aligns with recent design guidelines proposed for presentation-oriented visualizations Kosara [2016], advising specificity and compactness over generalizability.

3.1.3 Data Summarization Methods

Methods of data summarization can reduce the scale and complexity of a dataset for display in a summary. We specifically consider methods that summarize data while simultaneously provding a faithful representation of the underlying dataset. Prior work suggests methods

of re-organizing data for visualizations—for example, Card and Mackinlay [1997] argue that a small set of functions can be used to process data for visualization: filtering, sorting, multidimensional scaling, and selection by slider. Ellis and Dix [2007] taxonomize clutter reduction techniques for visualizations. Three of these techniques (sampling, filtering, and clustering) explicitly reduce data—however, their work considers summaries only as a means of reducing visual clutter in data space rather than emphasizing particular characteristics of the data.

We derive and propose four methods of data summarization from our observation and reconciliation of the literature: **aggregation**, **subsampling**, **filtering**, and **projection** (see §3.2.2 and §3.3.2 for details). These four categories capture the variety of methods that reduce data for display, and we anticipate that the method used will influence the types of judgments that viewers can make from the data visualized in the resulting visual summary (e.g., exploration of subsampling by Bertini and Santucci [2006]). Understanding the relevant tasks and judgments viewers will perform will help to connect these methods to their support in summary visualizations.

3.1.4 Tasks

The summarization methods used to summarize a dataset directly influence the analysis tasks supported by a derived summary. As an example of this relationship, using kernel density estimation to spatially aggregate values in a scatterplot helps viewers find dense clusters, but obscures local outliers. Our goal in this work is to collect a representative set of overview-level tasks, which capture the high-level information of a dataset. To do so, we look at the multitude of task taxonomies to generate a representative set of analysis tasks.

Task taxonomies have looked at how viewers obtain information from displays (see Andrienko and Andrienko [2006] and Shneiderman [1996] for canonical examples). Amar et al. [2005] identify a series of low-level tasks used to answer specific queries about a dataset. Ji Soo Yi et al. [2007] outline tasks that analysts perform to guide data interaction and exploration. Zhou and Feiner [1998] explore high-level presentation intents and visual discourse tasks, including "summarize" tasks such as *associate*, *compare*, *distinguish*, and *rank*.

More recent work considers how tasks can drive visualization design (see Rind et al. [2016] for a synthesis of this space). For example, Brehmer and Munzner [2013] looks at how tasks can be abstracted and expressed to support design across different application domains. Pike et al. [2009] look the mutual relationship between user tasks and interaction design. Schulz et al. [2013] describe how designers can reason about tasks using "5 W's" (and one "H"): why is a task pursued (a task's *goal*), how is a task carried out (a task's *means*), what does a task seek (the *target* and *cardinality* of objects), when is a task performed, and who carries out the task? Schulz et al.'s hierarchical synthesis of high-level tasks provides a representative analytic organization that we utilize in designing our codes for the survey. We additionally consider how these questions manifest in existing summaries to identify how tasks might guide effective summary design.

3.2 Methodology

We survey summary visualizations in the research literature to discover patterns in the use, design, and analytic affordances of visualizations using summarization. We are particularly interested in how the methods of data summarization are related to the use and the information communicated by the summary visualizations. To discover these patterns, we use four research questions to ground our exploration of this space. These questions concern the validity of our organization, how different types of summarization affect the resulting affordances communicated by the visualization, and how data and use affects how summarization methods are utilized. In detail, our questions are as follows:

Q1 Do the four proposed summarization methods cover the range of summarization performed for summary visualization design?

Q2 Does the method of data minimization affect how a resulting visualization can be used?

Q3 How does the use of summary visualization affect decisions of summary?

Q4 Does the type of data affect what types of minimization and affordances are appropriate?

To gather the necessary data to address these research questions, we use quantitative content analysis (QCA) [Riffe et al., 1998] that helps to quantify attributes about visual artifacts. This methodology has the advantages of quantiative evaluations (using statistical methods), and can break summary visualizations down into digestible factors to later identify trends between the factors. This in constract to grounded theory, which could build up concepts from qualitative exploration, but would likely be heavily biased by the sample of summary visualizations chosen. Instead, QCA depends on a static codebook to quantify attributes, evaluated by the coder. To promote ecological validity, we derive the codebook chiefly from existing visualization taxonomies (see §3.2.2), and use the results of the QCA process to validate our organization of summarization methods (**S1**). This methodology confirms the organizations proposed and the data generated through its use highlights trends in summary visualization.

Two data visualization researchers served as the coders for this survey. After a preliminary coding of ten papers, the two coders iterated on codebook definitions to clarify lingering ambiguities and to address emerging concerns regarding measure validity. Of the 180 evaluated manuscripts, 54 randomly-selected papers (30%) were redundantly coded for validation—the Cohen's kappa measurement for intercoder reliability found substantial agreement between coders ($\kappa = 0.71$, 86% overall agreement). Section 3.3 presents the result of this process, and identifies themes arising from our analysis.

3.2.1 Corpus Construction

To construct a corpus of summary visualizations, we use the data visualization research literature as a collection of peer-reviewed and valdated collection of visual analytic systems. This corpus is especially attractive due to the discussion of analysis scenarios in prose in close relation to the visual presentation of the summary visualization. To compose the corpus, we collected papers from the EuroVis, InfoVis, SciVis/Vis, and VAST conferences from 2009 to 2015 (1,158 papers). As coding each paper through a comprehensive census is untractable, we constructed a representative corpus of papers through simple random sampling from this set, as commonly used in traditional content analysis of large corpora (*cf.* Brubaker et al. [2012]). This resulted in a set of 180 coded papers (48 EuroVis, 53

InfoVis, 48 SciVis, and 31 VAST papers). Each paper was first coded for whether or not they included an implementation or technique that included a summary. Papers containing summaries were considered the artifact (figures and prose), and coded according to the presented protocol. We excluded theory, taxonomy, survey, toolkit, and evaluation papers as the focus of these papers was not on a single, specific visualization design, making application of the codes too subjective as we had no explicit evidence of the designers' intents.

Using examples from the visualization research community allows us to focus on designs whose quality, effectiveness, and utility have been reviewed by external experts in the field and that are tailored for a wide variety of applications. Although visualization designs are also found in conferences outside of the immediate visualization community (e.g., NIPS, VLDB, KDD), specific visualization contributions in these fields are relatively rare and unlikely to appear in a random sample. Further, visualization research papers emphasize novel contributions and techniques that represent the state-of-the-art in visualization specifically, and these papers represent a vetted corpus of summary visualizations that contain explicit rationales for their design discussed within the article, increasing the validity of our coding practices. However, the choice of this corpus biases the results of this study toward exploratory visualizations that are used by researchers or domain experts (not the general public), which we discuss in Section 3.4.

3.2.2 Coding Protocol

Each example in our 180 paper sample was labeled using a predetermined coding protocol designed to characterize four factors of summary visualization design: the visualization *purpose*, the *data* being used, the *data summarization* methods employed, and the *tasks* supported by the resulting summary visualization. We constructed our codebook by collecting and abstracting categories across 15 existing typologies describing data, purpose, task, and summarization in visualization. Table 3.1 summarizes the coding scheme used in the survey. As mentioned above, a set of 10 papers were used in iteration to revise our codebook for clarity. The final codes are as follows:

Category	Subcategory	Code
Purpose		Exploratory
		Confirmatory
		Presentation
Data Summarization		Aggregation
		Subsampling
		Filtering
		Projection
Task	Means: Navigation	Browsing
		Searching
		Elaborating
		Summarizing
	Means: Relation	Comparison
		Variations
		Relation-seeking
	Characteristics: High-level	Trends
		Outliers
		Clusters
		Frequency
		Distribution
		Correlation
Data		Data type
		Specific data
Other		Additional observations

Table 3.1: Two coders labeled 180 examples from the visualization literature. The coders first identified whether a visual summary was present and then coded each summary according to 22 attributes describing the summary's purpose, data summarization, and supported tasks (§3.2.2).

Purpose: We capture the purpose, or *goal*, of each visualization by considering whether it supports exploratory (undirected search), confirmatory (directed search) or presentation-oriented (exhibiting analysis results) analyses [Bertin, 1983, Schulz et al., 2013]. These codes describe the high-level intent of the summarization and are treated as three binary (present/absent) codes.

Data: We coded for data type using Shneiderman's data type taxonomy [1996], with one-dimensional and temporal data collectively coded as *sequence data*, encompassing one-dimensional data on a common axis (e.g., temporal, genomic, or ranked data). While we considered data size as a potential code due to the utility of summarization for large, multidimensional datasets, most systems did not provide specific information about the number of datapoints and dimensions tested and designed their methods for use with more than one dataset. For these reasons, coding for data size would have required significant assumption and extrapolation on the part of the coders, resulting in limited validity. We therefore did not consider data size in our analysis, but it is an important consideration for future work.

Data Summarization: We used our own observations coupled with categories from Schulz et al.'s *reorganization* task [2013], the visualization design space [Card and Mackinlay, 1997], methods for clutter reduction [Ellis and Dix, 2007], and methods of hierarchical abstraction [Elmqvist and Fekete, 2010] to propose a set of four data summarization methods used to reduce data for display:

Aggregation Computationally combining multiple elements (e.g., hierarchical aggregation [Elmqvist and Fekete, 2010]),

Subsampling Subsetting elements based on a stochastic selection of the data (e.g., random subsampling [Bertini and Santucci, 2006]),

Filtering Subsetting elements based on properties of the data (e.g., selecting a representative set [Ellis and Dix, 2007]), and

Projection Mapping data elements to a set of reduced or derived dimensions (e.g., principal component analysis [Jolliffe, 2002]).

We hypothesize that the data summarization methods used to construct a summary visualization heavily affect the affordances that the summary supports (**S2**). The method of data summarization was coded using a combination of four binary codes (present or absent for each of the above categories).

Task: Our task codes were drawn from the *means* and *characteristics* of Schulz et al.'s taxonomy [2013]. We chose this taxonomy as a general guide over other taxonomies (e.g., Amar et al. [2005], Brehmer and Munzner [2013], Casner [1991], Klein et al. [2006], Roth and Mattis [1990], Springmeyer et al. [1992], Ji Soo Yi et al. [2007], Zhou and Feiner [1998]) as it comprehensively reflected most categories presented in other taxonomies.

Means of navigation describe how summary visualizations support further analysis. These tasks coincide with Springmeyer et al.'s concepts of maneuvering [1992] and Casner's perceptual search operators [1991], and later used in Amar et al. [2005] and Ji Soo Yi et al. [2007] as intent in interaction, Zhou and Feiner [1998] in their modes of "enabling", and Heer and Shneiderman's "interactive dynamics" [2012].

Means of object-object relations describe information foraging tasks, including *comparison* (seeking similarities; see Gleicher et al. [2011], Ji Soo Yi et al. [2007]), *variation* (seeking dissimilarities; see Roth and Mattis [1990], Zhou and Feiner [1998]), *discrepancies* (seeking outliers Roth and Mattis [1990], Zhou and Feiner [1998]), and *relation-seeking* (seeking one of the aforementioned relations for individual objects; see Casner [1991], Heer and Shneiderman [2012]). While we additionally coded for *discrepancy*, this code was removed from our analysis due to poor agreement between coders.

High-level characteristics from Schulz et al. [2013] are used to code specific judgments of high-level data characteristics afforded by summary visualizations. While the source taxonomy does not explicitly define these characteristics, we used the following definitions that were agreed upon by the coders after iteration:

Trends	Estimate high-level changes across a dependent dimension,
Outliers	Identify items that do not match the modal distribution,
Clusters	Identify groups of similar items,

Frequency Determine how often items appear,

Distribution Characterize the extent and frequency of items, and

Correlation Identify patterns between dimensions.

These analysis tasks provide a representative proxy for understanding the informational utility of a summary visualization. Each of these three categories (summarized in Table 3.1) is measured as a combination of binary codes (task supported/unsupported).

Other: We recognize that a codebook constructed *a priori* may not capture all elements of designs and tasks of summarization. To capture traits of summary visualizations not captured by this initial set of codes, we allowed coders to note additional observations about each summary for further exploration.

3.3 Survey Results

Of the 180 papers coded, 104 (58%) contained summary visualizations. Of these, 64 (36% of those 180 total) provided enough detail within the paper to support valid coding across all four categories (described as *fully-coded summaries*). The remaining 40 consist of primarily scientific visualization systems focused on rendering, which provided little to no description of the target purpose or analytic tasks supported by the approach. To avoid over-extrapolation, we only coded those systems for data summarization.

By situating the 104 coded summaries within our design space, we identify factors leading to different design decisions, explore common design themes, and also understand aspects of this space that currently unexplored in visualization. In this section, we use our research questions to highlight significant findings from our analysis, and generate key *themes* (Table 3.2) that describe observations from the survey process. Taken together, these themes highlight core challenges for and opportunities for innovation in designing summary visualizations. These challenges concern how designers might exploit task specificity (addressing **Q2**), how systems leverage common design patterns (**Q3**), and how data affects design considerations (**Q4**). The full analysis results are available online at `http://graphics.cs.wisc.edu/Vis/vis_summaries/`.

Challenge	**Axis** — contributing factors	**Theme** — observations about the survey data	
Use	Purpose	T1	Summaries serve as a starting point for analysis.
(C1)	Purpose	T3	Confirmatory summaries support exploration.
	Purpose × Data Summarization	T5	Designs for communicating specific, known information use aggregation.
	Purpose × Task	T6	Summaries using subsampling emphasize exploration.
	Task	T15	Summaries act as roadmaps to guide detailed exploration by interaction.
Specificity	Purpose × Task	T2	Exploratory summaries encode a broad set of data characteristics.
(C2)	Purpose × Task	T4	Presentation summaries emphasize a small set of specific characteristics.
	Purpose × Data Summarization	T5	Designs for communicating specific, known information use aggregation.
	Data Summarization	T7	Most summaries use more than one data summarization method.
	Data Summarization × Task	T9	Summaries using aggregation support tasks characterizing the entire dataset.
	Data Summarization × Task	T12	Projection and filtering emphasize similar data characteristics.
	Data Summarization × Task	T14	Subsampling supports tasks that are statistically robust to random sampling.
	Task	T16	Summaries emphasize patterns that characterize all data and dimensions.
Design Patterns	Purpose	T1	Summaries serve as a starting point for analysis.
(C3)	Purpose × Data Summarization	T5	Designs for communicating specific, known information use aggregation.
	Data Summarization	T7	Most summaries use more than one data summarization method.
	Data Summarization	T8	Most summaries use aggregation to summarize data.
Data	Data Summarization	T10	Aggregation is common across all data types.
(C4)	Data Summarization	T11	Filtering can be used across all data types.
	Data Summarization	T13	Summaries using subsampling are most common for scientific visualization.

Table 3.2: Our analysis revealed sixteen common design themes in examples of summary visualization. Taken collectively as observations, these themes highlight the challenges in the design of summaries. We use these challenges to reason about the trade-offs in existing designs and to identify underexplored areas of the design space to inform new summary designs.

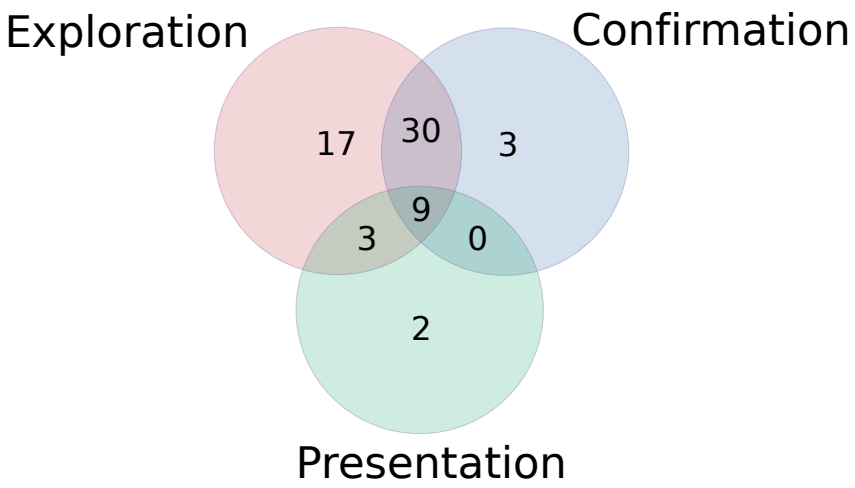


Figure 3.2: The distribution of summaries designed for each purpose over 64 fully-coded summaries (information visualization & visual analytics).

3.3.1 Purpose

Q3 addresses the question of how use of a summary visualization affects the design decisions of summary. To understand how use affects these decisions, we look at the data for statistical trends in the coded purpose of the visualizations. Purpose codified the intended use of summary visualizations for exploration, confirmation, or presentation (Figure 3.2). Most fully-coded summaries supported exploration (92%, 59 of 64), allowing viewers to analyze large collections of data without any *a priori* goals. 66% (42) of summaries were designed for directed analysis (confirmation), while only 22% (14) were explicitly designed to communicate known results (presentation). The dominance of exploration characterizes our first design theme: **summaries most frequently serve as a starting point for detailed analysis (T1)**. 95% (56 of 59) of these exploratory summary designs supported some sort of navigation task and 58% (34) allowed viewers to directly manipulate the granularity of the data encoded in the summary.

Additionally, **exploratory summaries support a broader set of data characteristic tasks (T2)**, such as identifying trends, outliers, clusters, frequency, distribution, and correlation. 70% (41) of exploratory summaries enabled viewers to explore more than three of the six high-level task characteristics (compared to 43% [6 of 14] for presentation) and 12% of summaries (7) supported all six. For example, Chen et al. [2016] uses a set of sum-

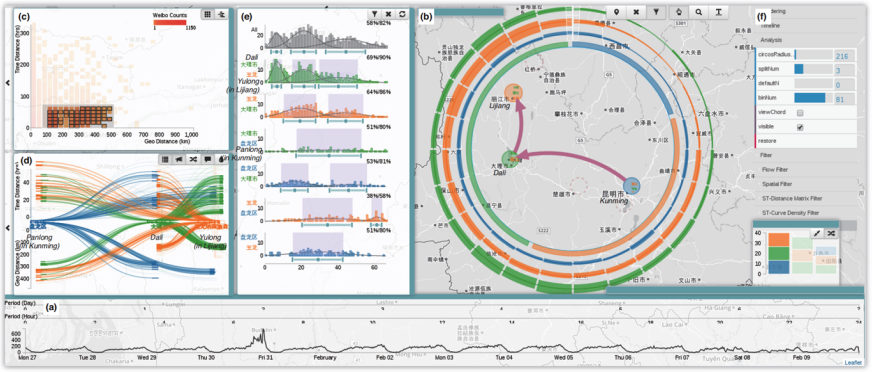


Figure 3.3: A visual summary in the system built by Chen et al. [2016] uses both aggregation and filtering in order to support a wide range of high-level analysis tasks.

marization methods to visualize different patterns across geo-tagged social media data. The resulting system (Figure 3.3) allows analysts to explore aggregate movement trends from social media data, and leverages interaction to enable analysis of the data distribution, frequency, and geospatial-based clusters.

Confirmatory summaries were often also exploratory: 61% of summaries (39 of 64) supported both exploration and confirmation while none were designed for confirmation or presentation alone. Like exploratory designs, confirmatory designs support a broader array of data characteristics than presentation-oriented summarization (68% supported more than three characteristics). This correlation suggests that **summaries designed for confirmation also support exploration (T3)**: confirmatory tools generally allow analysts to not only confirm specific hypotheses about data, but also to further refine and develop additional hypotheses about the data.

In contrast, **presentation summaries often emphasize a small set of data characteristics (T4)**. 57% (8 of 14) of presentation summaries communicated three or fewer coded data characteristics, and only one design communicated all six (Domino [Gratzl et al., 2014], which also supports exploration). All coded presentation summaries used aggregation to summarize data. Of these, 50% (7) used aggregation alone and 35% (5) used aggregation plus filtering. This suggests that **designs communicating specific, known information heavily rely on aggregation (T5)**. Aggregation can summarize data into a small number of precise features to emphasize known findings, encouraging effective presentation [Kosara, 2016]. This theme combined with **T2** highlights potential challenges in use and specificity

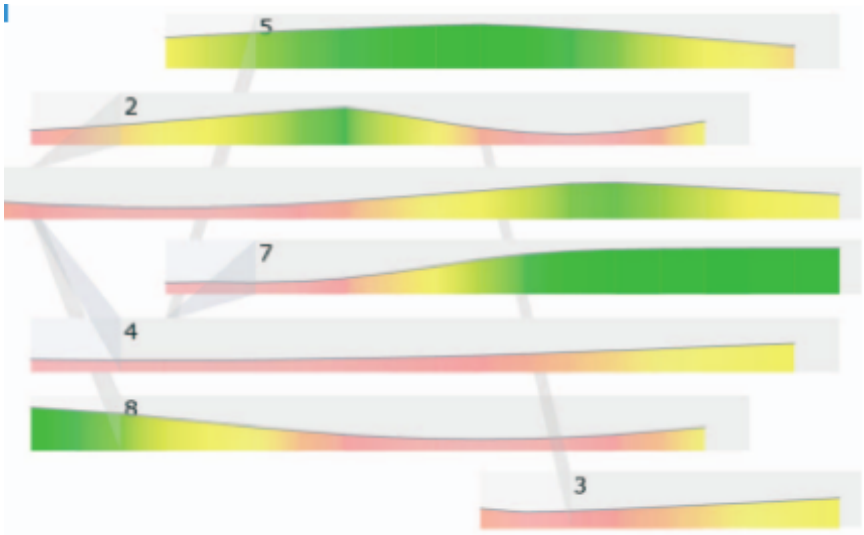


Figure 3.4: World Lines [Waser et al., 2010] aggregates spatial data across different simulation runs to allow viewers to directly search for the simulation with the best outcome.

(§3.4): focusing on specific properties of the data limits viewers' abilities to engage with data to better understand and evaluate presented findings whereas supporting many properties can overwhelm analysts or unnecessarily clutter a summary visualization.

Only five summaries were not explicitly designed for exploration. All five were confirmatory visualizations using aggregation, and none used subsampling. This bias indicates a trade-off between purpose and subsampling. **Subsampling methods favor exploration (T6)** as directed search may be inhibited by stochastically reducing data. Alternatively, aggregation helps guide analysts by presenting precise summarized values for well-defined tasks. For example, World Lines [Waser et al., 2010] uses aggregation to summarize parallel simulations of temporal events enabling comparison across known metrics for disaster planning (Figure 3.4).

3.3.2 Data Summarization Methods

All of the coded summaries employed at least one summary method (Figure 3.5), validating **Q1** that the organization is sufficient to cover the range of summarization operations. Unlike purpose and tasks, data summarization methods were coded for all 104 coded summary visualizations. We found that **most summaries used more than one data sum-**

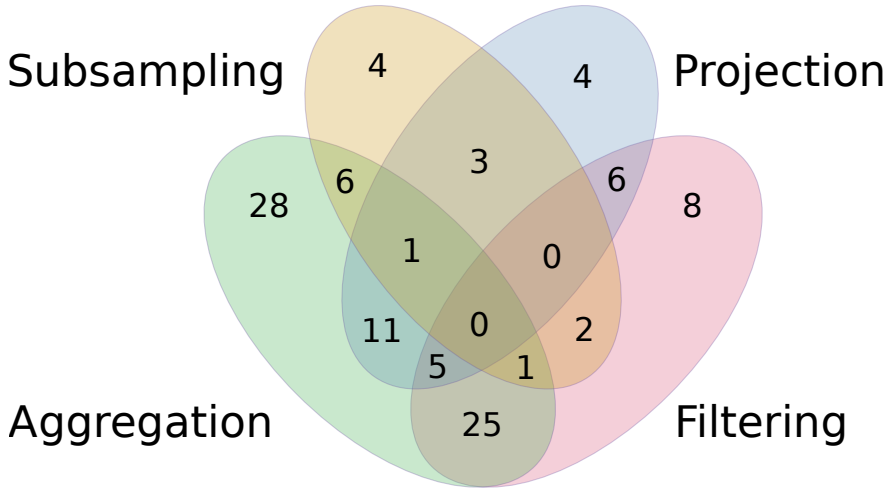


Figure 3.5: The distribution of summary designs using each data summarization method across 104 coded visual summaries.

marization method (T7) (63 summaries of 104, 61%), with 53 (84%) using exactly two. Each summarization method tended to favor a particular set of tasks. Combining summarization methods allows summary designs to leverage the strengths of individual techniques. However, there appear to be limits in how many summarization methods could effectively be composed: none of the coded summaries in our survey used all four summarization methods together. Rather, by understanding how each method is used in conjunction with other factors, we can analyze common design, use, and specificity patterns driven by these techniques. These observations are driven by **Q2**: how does the method of data minimization affect the resulting summary visualization?

Aggregation

Aggregation summarizes data by collecting and collating like-objects together through spatial, organizational, or attribute similarity. **Most surveyed visualizations (74%) use aggregation to reduce data (T8)**, with 27% exclusively using aggregation. **Visualizations frequently used aggregation to support tasks characterizing the entire dataset (T9)**. Of the 64 fully-coded examples, aggregation frequently supported both distribution (42 of 54, 78%) and clusters (43 of 54, 80%).

Visualizations often used these methods of data reduction to take advantage of trade-offs between aggregation and filtering: while aggregation emphasizes characteristics describing

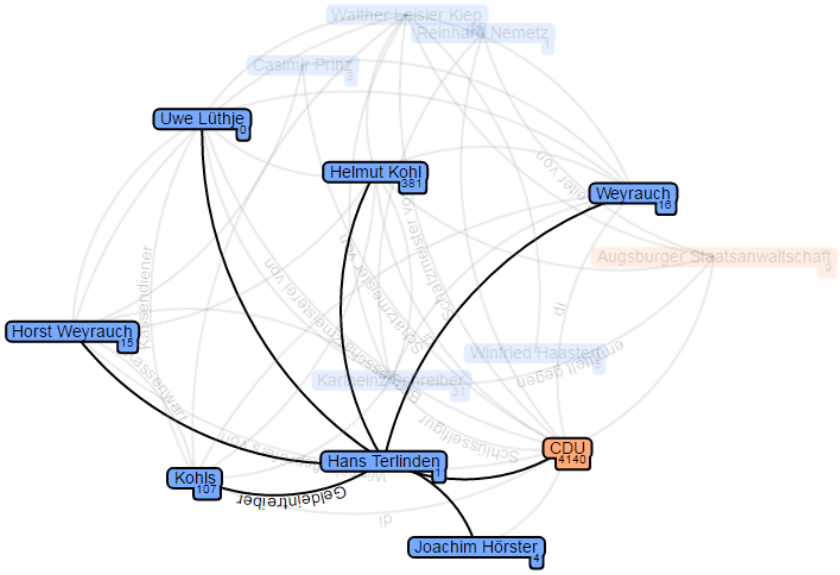


Figure 3.6: Most network summaries, such as Networks of Names [Kochtchi et al., 2014], combine aggregation and filtering to summarize data. The system aggregates different relations across pairs of entities and filters these patterns according to their frequencies to encode the relationships that best characterize the dataset.

multiple datapoints, filtering can help tailor these characteristics towards interesting or relevant collections. For example, the Network of Names [Kochtchi et al., 2014] first aggregates recurring relations in social networks and then filters out uncommon relations to emphasize dominant patterns in large actor networks (Figure 3.6). Filtering can also be used to reintroduce important data values obscured through aggregation, such as outliers in a scatterplot aggregated by density [Mayorga and Gleicher, 2013] (Figure 3.7). We found that summarization without aggregation targeted these kinds of individual value judgments, such as identifying outliers which was supported by 70% of non-aggregate visualizations.

In terms of **Q4** (effect of data type on summarization), **aggregation was commonly used across all data types (T10)**, occurring in more than half of the surveyed papers across all data types. The dominance of aggregation across all data types indicates that it is a "default" used in visualization systems. Although aggregation is a powerful technique, it communicates specific properties of a dataset at the cost of the underlying data values. To use aggregation effectively, designers must know what properties of the data are important to the user and how to compute and encode these properties to faithfully represent the underlying data. Summaries using aggregation exchange flexibility for specificity, and crit-

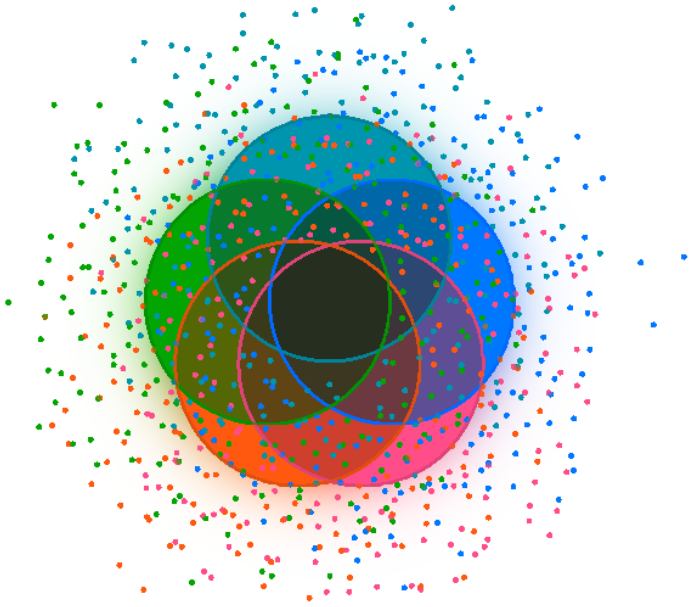


Figure 3.7: Splatterplots [Mayorga and Gleicher, 2013] represent two-dimensional points by combining a kernel density estimation with filtering and subsampling of representative outlier points. Combining aggregation and filtering takes advantage of the trade-offs between these methods to support a broader variety of tasks.

ically examining this trade-off may offer new opportunities for visualizations—discussed in detail in §3.4.

Filtering

Filtering is commonly used to allow analysts to specify meaningful properties of the data or compute representative subsets. 47 visualizations (44%) used filtering; however, filtering was seldom used in isolation (17% of all filtering summaries, supporting **T7**). Filtering in visualizations allowed analysts to identify clusters (23 of 47 filtering visualizations, 82%), characterize distributions (22 of 47, 79%), and evaluate correlation (17 of 47, 61%). Filtering tended to reduce extraneous data to support and highlight these types of high-level judgmets, reflecting the visual information seeking mantra [Shneiderman, 1996]: "overview first, zoom and **filter**, then details on-demand." Filtering in these cases helps analysts find interesting subsets of the data to explore.

Like aggregation, **filtering supported summary designs for all data types (T11)**. However, visualizations leveraging filtering provided analysts with little information about how

filtering for these properties might bias potential interpretations, again raising challenges for designers around summarization specificity.

Projection

As a method of summarization, projection is used to re-organize data as part of several summarization operations. 30 examples (28%) used projection to summarize data, with most projections summarizing large collections of documents (7 of 30, 23%), 3D fields (9 of 30, 30%) and multi-dimensional datasets (10 of 30, 33%)—highlighting the utility of projection for high-dimensional data (**Q4**). Similar to filtering, projection was seldom used in isolation (**T7**), and was commonly paired with either aggregation or filtering (24 summaries, 80%). For example, text documents can use topic modeling to project document vectors into a lower dimensional space and aggregate documents according to these topics (e.g., Cui et al. [2014]).

Projection-based summarization emphasizes similar data characteristics as filtering (T12): locating clusters (17 of 19 summaries, 89%), characterizing distributions (16, 84%), and evaluating correlation (14, 74%). However, projection frequently also enabled outlier analysis (15, 79%). Visualizations can combine filtering and projection to help highlight critical patterns in complex data. For example, Progressive Insights [Stolper et al., 2014] projects patterns onto statistical axes and filters the strongest patterns along each axis to highlight the strongest patterns over each new dimension.

Regarding **Q3** in the use of summarization methods, we found that projection was seldom used for presentation (2 of 20, 10%), but instead supported in-depth explorations, as in Progressive Insights. We hypothesize that this is because the mathematical complexity of many methods make it difficult to clearly communicate meaningful narratives about the data, leaving designers to reason closely about use and specificity when using projection techniques (§3.4). However, we acknowledge that this may be biased by our choice of corpus, as we discuss in Section 3.4.

Subsampling

The act of subsampling reduces data for display by stochastically and indiscriminately removing objects from the dataset. While relatively few visualizations used subsampling to reduce data (16% of the 104 sampled), subsampling is also commonly paired with another summarization method (aggregation: 8 visualizations, 47% of subsampled examples; filtering: 3, 18%; and projection: 4, 24%). Similar to projection, subsampling is commonly used as a conjunctive operation to reduce data to manage the complexity of the resulting visualization.

Subsampling was predominantly used for spatial visualization (T13) (11 of 17 examples, 65% of subsampling use), where it reduced the visual complexity of aggregated structural data. Only six fully-coded visualizations used subsampling. These visualizations primarily support trend analysis (5 summaries, 83%) and characterizing distributions (5, 83%). These high-level characteristics indicate that **subsampling can support summarization where the analysis tasks are statistically robust to random sampling (T14)**. While few summaries use subsampling in practice, it is the only data summarization method that does not bias the resulting summary towards any specific attribute of the data. This implies subsampling may be a powerful tool for summaries for novel exploratory visualizations, especially when the target tasks or properties of interest are unknown *a priori*.

3.3.3 Tasks

While several of our prior design themes address relationships between methods of data minimization (**Q1**, **Q2**), and data types (**Q4**), we also explore how the utility of summary visualizations affect design decisions (**Q3**). The trends here help to inform how summarization affects analytic trade-offs in visualizations (Figure 3.8). From the 64 fully-coded visualizations, we found themes around how designs allow viewers to navigate the dataset, how summarizing different data types prioritize different analyses, and characteristics of the data that summarization universally preserves.

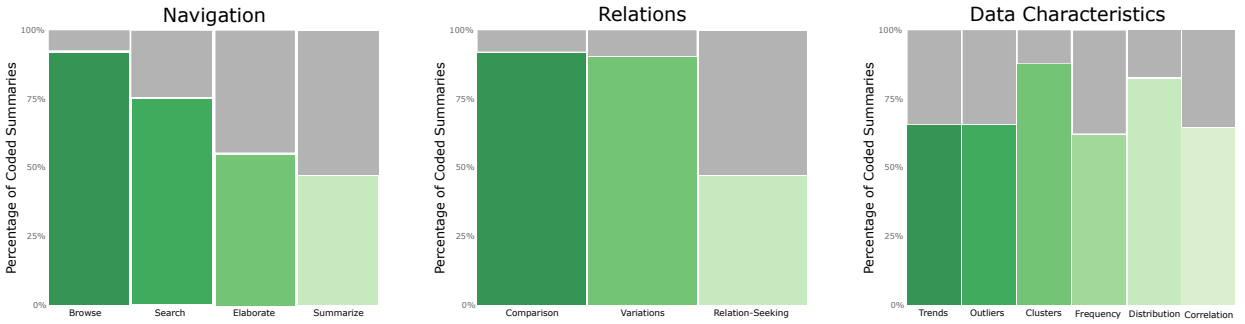


Figure 3.8: The distribution of summary designs supporting different kinds of analysis tasks across 64 fully-coded summaries.

Means of Navigation

Most visualizations presented summarized data to allow browsing for unknown patterns in data (58 of 64, 91%) while a smaller number supported directed search for known patterns (48 of 64, 75%). Among those, 13 visualizations (20%) supported browsing but not searching. These summary methods tended to emphasize relationships across collections of datapoints: all but one emphasized both clusters and outliers, and all but two communicated value distributions. For example, in Brehmer et al.'s juxtaposed matrix and faceted box plots [2016], the aggregate matrix obscures local patterns to prioritize aggregate temporal clusters while box plots encode distribution and outliers (Figure 6.2). This aggregation prevents directed search for individual motifs; however, the interaction between box plots and matrix cells allows viewers to browse for interesting local patterns. This exemplifies how **effective summaries can act as roadmaps to guide user interactions with the data (T15)**. This raises an important challenge for visualization designers to consider when summarizing data: what properties of the data might make for an effective starting point?

Our survey revealed that most designs start with the most abstract available data representation, then allow analysts to drill down into the data to uncover specific details. Many summaries did not allow viewers to change the level of detail without changing the visual representation (28 visualizations, 44%). All of these visualizations used additional supplemental designs to support detailed exploration, supporting **T15**. For example, glyph SPLOMs [Yates et al., 2014] summarize distributions within SPLOMs so viewers can identify scatterplots to explore in detail (Figure 3.10).

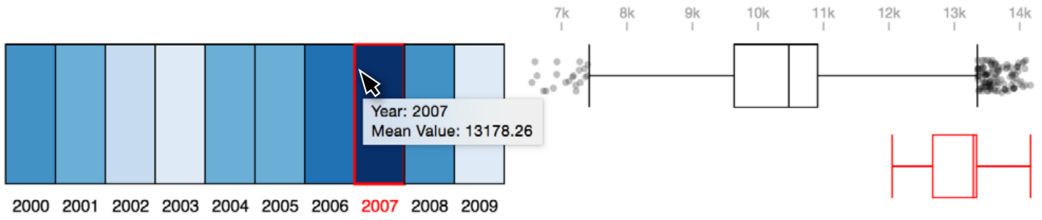


Figure 3.9: Summaries supporting browsing, but not directed search, tended to emphasize properties of collections of datapoints, such as distributions and clustering. For example, Brehmer et al. [2016] use aggregation allow viewers to identify high-level temporal clusters, outliers, and distributions and use interaction to browse for interesting underlying distributions; however, this aggregation obscures smaller scale motifs, preventing viewers from localizing specific patterns.

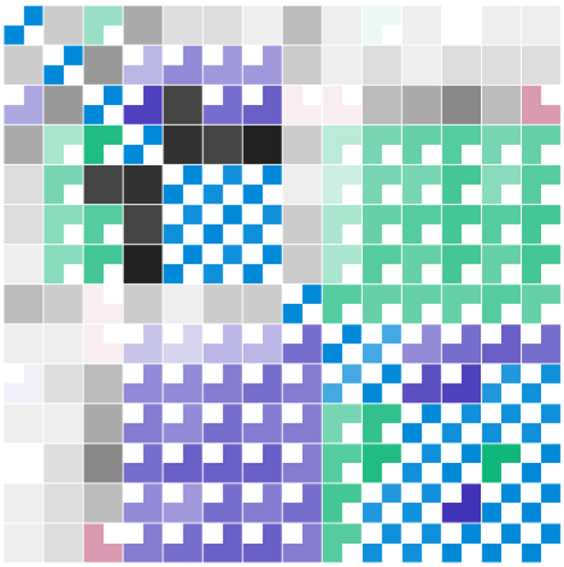


Figure 3.10: Summaries act as roadmaps for exploration, starting with a high-level of abstraction and often requiring viewers to use alternative representations to explore details. For example, glyph SPLOMs [Yates et al., 2014] summarize the quadrants where datapoints are clustered in each scatterplot of a SPLOM. Viewers can then look at specific scatterplots to explore interesting data in more detail.

Means of Relation

Most visualizations used summarization methods that enables viewers to identify similarities (89%) and differences (88%) between collections of datapoints. However, significantly fewer support relation-seeking between individual items (45%), with most of these being network visualizations. Network visualization for relation-seeking use some combination

of aggregation or filtering (2 aggregation, 2 filtering, and 4 aggregation+filtering). The correlation between network data and relation-seeking implies that summarizing network data often requires emphasizing relationships between key portions of the network. Aggregation (by collapsing important collections of nodes or edges) and filtering (by preserving meaningful or common relations) allow designers to meaningfully summarize networks. For example, Networks of Names [Kochtchi et al., 2014] highlights relationships between large collections of entities by first aggregating all entity relations and then filtering on these aggregate frequencies to visualize the most common relations in the dataset (Figure 3.6). The only coded network visualization that did not rely exclusively on aggregation and filtering, SAVE [Shi et al., 2011], did not emphasize relation-seeking and instead focused on multidimensional measures associated with each node. Despite the structural similarities between network and hierarchical data, the latter tended not to support relation-seeking though more visualizations of hierarchical data should be explored before drawing conclusive insights.

Data Characteristics

Summarization most frequently preserved characteristics that describe the entire collection of data and dimensions: clusters (80%) and distributions (75%). Trends (59%), outliers (59%), frequency (56%), and correlation (58%) were roughly equally supported across all visualizations. The bias towards clusters and distributions suggests that **summarization often emphasizes descriptive aggregate patterns across all of the data and dimensions (T16)**, rather than patterns in individual values or relationships between specific dimensions. 11% of coded summary visualizations support all tasks (7 of 64).

We found a bias towards particular task affordances and summarization methods across data type (addressing **Q4**). For one-dimensional data, many visualizations support discovering clusters (9 of 10, 90%) through aggregation (9, 90%). For 2D data, many visualizations support discovering trends (7 of 8, 88%) and frequency (6, 75%). In comparison, 3D data summarization tends not to support trends or frequency judgments (3 and 1 of 7, 43% and 14%, respectively), but instead preserves distributions (5, 71%). Neither multidimensional nor network data used subsampling (5 summaries of 24, 21%; 0 of 10, 0%; respectively). We

anticipate this bias arises from stochastically removing information that could potentially remove critical structures in the data, such as relations between different levels of hierarchy or across different data dimensions.

3.4 Discussion

Our four research questions lead to observations identified through the QCA process, resulting in 16 design themes (Table 3.2) of summary visualizations in visual analytics. Through the survey process, we confirm that the four methods of summarization are sufficient to encapsulate data re-organization for display in a visual summary (**Q1**). Here we describe these challenges and opportunities in how viewers *use* summaries, and in how designers consider *specificity* in data summarization, leverage common *design patterns*, and tailor summaries to specific *data*.

3.4.1 Use (C1)

We address the questions of how the use of summary visualizations affects design choices (**Q2** and **Q3**) through the following observations. Summary visualizations frequently serve as a starting point for analysis (**T1**), providing a roadmap for detailed exploration using alternative views or interactions (**T15**). To help guide analysis, designers often choose an summarization method and target characteristics based on a visualization's intended *use*: how does the summary guide subsequent interaction and interpretation? To tell an immediate and focused story (e.g., presentation), summarization emphasizes specific patterns (**T4**) while open-ended analyses are better supported by summarizations encoding a broad set of characteristics (**T2**).

Challenges: Addressing the use of exploratory summary visualizations (**Q3**), these visualizations generally present many data characteristics at once, which offers analysis flexibility but might also overwhelm viewers: they may not know which questions to ask first. Exploratory summaries might instead choose to depict subsets of important characteristics to guide viewers through a more targeted analysis sequence. This targeting could

be especially beneficial for domains with established analysis workflows or for guiding novice analysts who could become overwhelmed when faced with too much information.

Opportunities: Summary visualizations that violate assumptions around use may offer interesting trade-offs. For example in response to our research question on the use of summary visualizations, we found that summarization for presentation generally targets a smaller set of data characteristics, whereas exploration supports a larger set. On the surface, this pattern makes sense: presentation tells a story, while exploration searches for unknown patterns in data. However, inverting this pattern may be advantageous. While aggregation can communicate specific information, explicitly visualizing statistics about the data may cause viewers to misinterpret secondary characteristics [Correll and Gleicher, 2015]—for example, trend lines can cause viewers to too liberally label outliers. In response, designs using filtering or subsampling may alleviate potential biases and better familiarize viewers with the data. Further, allowing access to more data properties can allow viewers to construct their own interpretation of the dataset in the context of the arguments made through the visualization.

3.4.2 Specificity (C2)

One of our core questions is how the method of data summarization affects the types of information communicated by a resulting summary visualization (**Q2**). Existing summaries heavily emphasize data characteristics that describe datasets in aggregate rather than specific data points or dimensions (**T16**), using aggregation methods to compute and visualize specific patterns in data (**T5**). However, aggregation explicitly tailors summarized data to specific statistical tasks, visualizing a computed representation rather than the actual raw data. In contrast, subsampling might remove datapoints that are important to a particular story, but also reduces clutter and potentially denoises data while providing immediate access to the underlying data (**T6**). This trade-off characterizes summarization *specificity*: aggregation can target specific high-level data characteristics but obscures specific values, whereas subsampling and filtering encode individual data values but rely on viewers to estimate characteristics.

Challenges: We found that existing systems favor specificity over data fidelity. Even if important data characteristics are not known *a priori*, aggregation was often used to express generic properties of the entire dataset (**T9**), such as distributions and clusters (**T7**). Filtering, subsampling, and projection are seldom used without aggregation; however, designs using these methods frequently preserve the underlying characteristics of the data (**T12** and **T14**).

Our results identify a need to more carefully consider how the broad use of aggregation may bias analysts. Aggregation generally focuses on precisely encoding a specific set of characteristics at the expense of allowing viewers to synthesize their own perspectives from available information. When favoring specificity, designers must carefully consider how their summarizations influence the interpretation of the data, especially as summary visualizations are frequently the first thing that analysts encounter when exploring their data (**T1**).

Opportunities: Favoring breadth over specificity supports serendipitous exploration of summarized data. Designing for serendipty can foster new discoveries or generate unexplored hypotheses [Thudt et al., 2012b] by broadly supporting a plethora of tasks. Subsampling, the least common summarization method in our survey, may be especially helpful in designing for serendipity: subsampling summarized data are statistically unbiased against properties of the dataset. It provides designs with low specificity, but generally preserves aggregate characteristics of the data. Further, stochastic sampling can create summaries that are not subject to the same confirmation biases as targeted filtering or aggregation.

Summary designs should also consider how designs let viewers combine information through visual aggregation. This understanding and explicit use of visual aggregation is just emerging in the visualization literature (see Szafir et al. [2016] for a survey), and our random sample did not identify any summaries explicitly designed for visual aggregation (e.g., Sequence Surveyor [Albers et al., 2011]). However, visual aggregation may allow designers to tailor summaries to specific tasks while using summarization methods. A better understanding of how visual aggregation factors into the specificity of designs is important future work.

3.4.3 Design Patterns (C3)

Several design decisions were reflected in the majority of the coded summary visualizations (Fig. 3.8), helping to address the research question of how summary use affects design decisions (**Q3**). Understanding these seemingly "default" decisions can guide novel design thinking for summary visualizations, as well as proposing good, starting design foundations. For example, almost all surveyed systems used more than one data summarization method (**T7**). Compositing summarization methods can emphasize particular data characteristics and increase the number of tasks supported, but has the potential to increase the distance between the representation and the semantics (structure) of the original data. Aggregation, for example, is most commonly paired with other methods (**T8**), but aggregation techniques are often data-dependent and require viewers to interpret computationally transformed data. In these designs, using multiple summarization techniques to increase task support comes at the expense of usability: the viewer must perform more mental processing to translate visual patterns back to the underlying dataset.

Challenges: The use of design patterns in summarization encourages reproducibility and reduces the analyst's overhead in learning new systems. However, designers must consider whether a particular design pattern is appropriate given the data type and analysis goals. To date, no concrete guidance exists for understanding design pattern effectiveness. Our results indicate a need to collect and standardize design patterns and evaluate their potential utility. Our design space provides a preliminary scaffold to build this knowledge.

Opportunities: A common design pattern was the use of summary visualizations as a starting point for exploration (**T1**). While this pattern aligns with conventional visualization guidelines [Shneiderman, 1996], designers might also consider how an analysis might craft a summarization to serve as ending point for an analysis. Insights from exploratory visualizations are often constructed longitudinally, building up as viewers learn more about their data [Saraiya et al., 2006]. Summarizations might arise as descriptors of the insights constructed during an analysis. While no surveyed summaries enabled this inductive summarization, a few visualization systems incorporated annotation within a summary component in order to iteratively refine overviews from insights (e.g., TenniVis [Polk et al.,

2014] and Overview [Brehmer et al., 2014b]). For example, Overview lets analysts label and manipulate datapoints to construct understanding across documents. Considering how designs might support summaries as generative artifacts of an analysis, capturing features like provenance, model refinement, and insight development requires moving away from default design patterns to inspire new summarization capabilities and applications.

3.4.4 Data (C4)

Q4 addresses the question of how the data type affects the affordances of a summary visualization. Several themes highlight patterns between specific data types and summarization choices. In some cases, these patterns help guide particular designs. For example, summary visualizations use aggregation and filtering for any data type (**T10** and **T11**), whereas subsampling is generally used for spatial datasets (**T13**). Designers may be able to use common patterns across data types to better reason about how summarization methods might support heterogeneous data, as well as how to adapt summarization techniques across domains.

Challenges: The semantic and statistical properties of the underlying dataset and analysis goals can limit candidate summarization methods. For example, continuous 2D data can be meaningfully summarized using kernel-density estimation (KDE), whereas a kernel does not easily map to hierarchical data. We identified some voids in the factors for particular data types, including lack of frequency support for summaries of three-dimensional data, and a lack of subsampling examples for network and multi-dimensional data. These voids identify places where innovative methods are needed for intuitive summarization.

Opportunities: Specific data types tended to favor specific summarization methods. For example, summaries of document collections and scientific data rely heavily on projection (**T13**). Designers can use this correlation to derive design inspiration in other domains: how might projection effectively summarize datasets that are structurally similar to documents, such as collections of event sequences? An important aspect of understanding and applying our design space in practice will be understanding how different summary approaches might generalize across data types and domain scenarios.

3.4.5 Limitations & Open Questions

This work begins to answer our research questions, taking preliminary steps towards a broader discussion of data summarization in visual analytics. However, our data-driven approach is inherently limited by sampling. Although we anticipate that the collected systems and themes characterize summarization more broadly, we cannot make absolute claims about the generalizability of our results. Instead, this work allows us to identify challenges and opportunities for visualization design that will help extend and enhance a more principled use of data summarization. For example, our observations identified several patterns in summaries designed for particular data types, but our sampling across different data types is limited. Future work could provide deeper coverage across different data types through stratified random sampling to identify biases across different designs. This could inspire both generative guidance for summarizing data across domains and novel design techniques for guiding innovative summarization techniques.

Our dataset is also biased towards exploratory visualizations, which is likely a function of an underlying bias in the visualization research literature [Kosara, 2016]. While we elected to use this literature to ground our coding in the design intents of the authors (§3.2), summaries from other sources, such as data journalism, could help create guidelines that inform summaries for a larger variety of practitioners and uses.

Our observations from this survey begin to answer the four proposed research questions. The four methods of re-organizing and summarizing data (**Q1**) are confirmed as being sufficient—every example of a summary visualization was matched into one or more methods. We identified how methods of minimization affects the resulting affordances and use of visualization (**Q2**), including observations such as how subsampling tends to support tasks dealing with data characteristics that themselves are resistant to missing data (**T14**). We observed that the use of summary visualizations affected design decisions (**Q3**)—as an example, aggregation is used to focus the viewer for presentation tasks (**T5**). Lastly, we identified overrepresentation of summary methods for particular data types (**Q4**), such as how subsampling was most used for spatial datasets (**T13**). The observations from the survey helps to create a clearer picture for the design of summary visualizations.

3.4.6 Conclusion

As datasets grow in size and complexity, effectively leveraging summarization becomes increasingly critical for visual analytics systems. We crafted a design space for summarization and used this design space to evaluate 180 papers from the visualization literature using QCA. Our analysis identified the importance of summarization for visualization (employed in 59% of surveyed manuscripts) and 16 design themes relating visualization purpose, data summarization methods, data types, and analysis tasks. We found trade-offs in the use of different summarization methods and biases in their applications in existing designs. These themes highlight patterns in the design for summarization that can guide viewers using visualization systems.

This work is a critical step in characterizing a design space of summarization and creating a set of design patterns for summary visualization. Our four research questions help to validate the proposed organization of summarization methods (**Q1**), and identify over- and under-represented trends between the factors of purpose, summarization, affordances, and data type (**Q2–4**). As a result of this process, these observations comprise a foundation based in realized, visualization design. This foundation provides a base set of guidelines in designing bespoke summary visualizations, and also suggests some potential design defaults for interactive, viewer-centric visualization systems.

As a result of this work, we identify four methods of reducing and re-organizing data for summarization. We have shown that these four methods tend to select the types of high-level information that can be obtained from the resulting visualization. Through a systematic random sample of the visualization literature, we can obtain trends in summary visualization design and use, and highlight correlations that appear. In the following chapter, we explore how different design methods for summarization manifest themselves in a scatterplot design paradigm, and create a framework for understanding what factors (data characteristics and tasks) make some summarization designs more appropriate than others.

4 DESIGN FACTORS FOR SCATTERPLOTS AT SCALE

In this chapter, we focus on a single type of common visualization: the scatterplot. By constraining our design focus to this visualization type, we can generate an actionable organization to analyze how different factors affect the appropriate design of a scatterplot. Such a framework serves not only to help practitioners and designers select appropriate designs for the desired use case for data at scale, but also helps to organize contemporary techniques to identify alternative designs and areas of underexplored design solutions. The abstractions developed here can be adapted for a visualization practitioners' benefit—by using the factors of task and data characteristics, a host of predetermined appropriate designs and techniques can be proposed (more detail in Chapter 7 and 8). Following the design factors proposed in Munzner [2014], we create the first such design framework for scatterplots. A publication representing the work in this chapter was published as part of the 2017 InfoVis proceedings [Sarikaya and Gleicher, 2018].

4.1 Overview

Scatterplots are a very common type of visualization. Their flexibility has led to their use in a variety of exploratory and presentation contexts. The traditional scatterplot represents each object in a dataset with a point (or other mark), positioned on two continuous, orthogonal dimensions. As data grows in scale and complexity, the traditional scatterplot design rapidly becomes ineffective. As a result, many other scatterplot designs have been proposed. While these designs may address scale, they are often specific to data characteristics and tasks. Designers have little guidance in how to select among design choices. Our goal is to help designers select scatterplot designs that are appropriate to their scenarios by identifying the factors that affect the appropriateness of scatterplot designs.

In this work, we describe how to consider analysis scenarios in terms of their task and data characteristics in order to determine which scatterplot designs are appropriate (see Figure 4.1). We generate a framework by collecting and abstracting use cases of scatterplots in the literature. For *tasks,* we collect model tasks that are performed with scatterplots,

creating an abstraction that helps us to understand the task space of scatterplots. We also identify a number of design-relevant *data characteristics*, such as the number of objects. To identify the space of potential *designs*, we survey scatterplot designs to organize and cluster similar design decisions together. We use tasks and data characteristics to reason about the applicability of these designs. Our framework, therefore, provides a process for designers to select scatterplot designs appropriate to their scenario by first identifying relevant task and data characteristics. Additionally, the framework highlights areas in the design space for further exploration, and where multiple solutions exist for similar, abstract problems.

The framework that we construct in this chapter uses analysis task and data characteristics to identify the scenarios in which a design is appropriate, much like the methodology championed in Munzner [2014]. Through this chapter, we will summarize a short history of the scatterplot and related designs (§4.2.1), orient our framework relative to existing visualization taxonomies (§4.2.2), frame and identify the relevant factors that affect scatterplot design (tasks [§4.3] and data characteristics [§4.4]), survey the space of designs (§4.5), and explore how the framework and its factors can be used to determine the appropriateness of scatterplot designs (§4.6–4.7).

4.2 Background

4.2.1 Scatterplots

The scatterplot was designed to emphasize the spatial distribution of data plotted in two-dimensions. While the scatterplot itself has had a long history (see Friendly and Denis [2005]), its relative simplicity and flexibility enables the scatterplot as an ideal sandbox for early information visualization and perceptual psychology research. In particular, Cleveland [1985] notes three factors that may affect the design decisions that are made by the designer of a scatterplot: (1) marks or points are designed with preattentive features in mind, (2) scatterplots are designed with the detection of individual objects in mind, and also (3) are designed such that the distances between objects represent a notion of similarity.

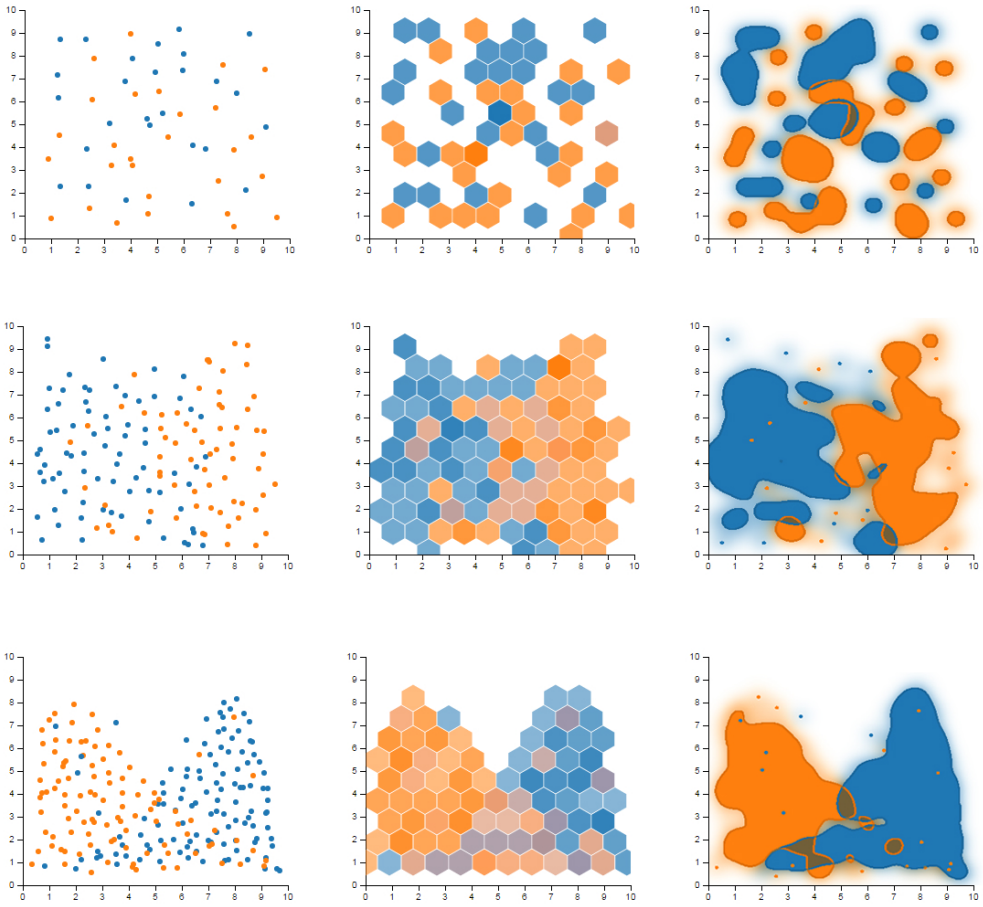


Figure 4.1: Scatterplot designs (shown in columns) have varying levels of support for viewer tasks based on the data characteristics (rows). Here, we compare a traditional scatterplot (left column) to a hexagonal binning implementation [Carr et al., 1987] (middle) to a Splatterplot [Mayorga and Gleicher, 2013] (right) for three representative datasets. The appropriateness of a scatterplot design is based on the characteristics of the data and the design's support of the viewer's task (such as identifying outliers or comparing distributions). For random distributions with few points (top row), the traditional scatterplot (left) describes the data plainly. With increasing numbers of points (middle row), aggregation representations such as binning (center) communicate spatial density. With overlapping distributions (bottom row), density-based representations communicate overlap and can also show outliers (right), which disappear in the binned representation (middle).

With different sets of guiding factors, many different variations around the core scatterplot design have been developed—many trying to squeeze more fidelity from the traditional mark-per-object, two-dimensional scatterplot design. These designs are typically at odds with factor (2) above, prioritizing aggregate judgments over object-centric affordances. In this section, we highlight the background of challenges in adapting scatterplots to different

analysis scenarios. While some of these strategies utilize multiple scatterplots, linked components, or glyphs as marks, we only consider the use of a single, two-dimensional, scatterplot with mono-variate marks outside of this section for the purposes of concision.

Dealing with too much data

Scatterplots work very well for a variety of analyses—until the amount of data overwhelms the traditional design of assigning a mark to every datum in a dataset. *Overdraw* is a common concern for scatterplots, defining the scenario where marks overlap one another and mask marks drawn under them. Cui et al. [2006] notes that the drawing order can have serious ramifications of emphasizing inaccurate judgments of distribution. Fekete and Plaisant [2002] highlight technical issues in displaying millions of items, where overdraw is a prime concern.

Reducing the data is one approach to address the challenge of too much data. Strategies include reducing the data before mapping to a visual representation, simplifying the visual representation itself, or modifying the space of the plot. In the first case, stochastic or stratified subsampling of the data [Bertini and Santucci, 2006, Chen et al., 2014] is an example of reducing the number of data for display. Binning data [Carr et al., 1987] by collecting counts within small localized regions and visualizing area-aggregated, relative counts is another strategy in this same vein. Strategies to simplify the visual representation, such as continuous density estimation used by contour plots [Collins et al., 2009], landscape maps [Tory et al., 2007], and Splatterplots [Mayorga and Gleicher, 2013], aggregate marks by their position, highlighting clusters and distributions of marks.

Modifying the space of the plot can also emphasize hidden structures. Generalized scatterplots [Keim et al., 2010] and related work (e.g., continuous scatterplots [Bachthaler and Weiskopf, 2008]) take advantage of open space in a plot by performing a subspace warp to take advantage of unused regions of the graph, while combining the strengths of density estimation. In addition to these techniques, organizations of the strategies have been proposed, most notably Ellis and Dix [2007] on clutter reduction where many strategies are directly applicable to scatterplot data. In this chapter, we provide organization of the factors specific to scatterplots that can assist in selecting these types of design elements

and techniques from the possible space of all designs.

Dealing with high-dimensional data

Scatterplots have enjoyed continued use in the visualization of high-dimensional data. Brehmer et al. [2014a] outline some of the analysis scenarios covered by scatterplots and related visualizations. Using scatterplots, the three common strategies are to select a subset of two dimensions, reduce the dimensionality to two dimensions using a dimensionality reduction technique, or showing all dimensions in a pairwise fashion. In the first case, simply showing a subset of two dimensions reduces to the typical scatterplot use case, though the distance between marks only communicates similarity in a reduced subspace.

Commonly, dimensionality reduction methods use scatterplots to visualize their results. Techniques may project points using such a method to cluster similar objects together, such as the work by Lehmann and Theisel [2013] and Yuan et al. [2013]. Some other scatterplot-related designs bridge the gap back to feed input back to dimensionality reduction techniques, such as Dis-Function [Brown et al., 2012] and InterAxis [Kim et al., 2016] by using direct manipulation to drive and update object clustering and projection functions.

SPLOMs [Carr et al., 1987] are a popular choice for visualizing pairwise dimensional information, highlighting correlations between pairs of dimensions. However, the paradigm does not scale well to high numbers of dimensions. In response, scagnostics [Wilkinson et al., 2005] provide metrics for identifying interesting correlations and patterns in two-dimensional data, including features described as shape, trend, and coherence. These measures can be used to find interesting combinations of dimensions to visualize, as shown in both Bertini et al. [2011] and Tatu et al. [2010], and can be used to help guide interaction, as shown by Dang and Wilkinson [2014]. To support increasing complexity in high-dimensional data, there have been variations on the SPLOM and scagnostic themes, including the use of radial graphs [Kandogan, 2001] to show all dimensions in a two-dimensional plane. Yates et al. [2014] takes an additional step of abstracting the "shape" of pairwise correlation in individual scatterplots within a SPLOM, highlighting trends of correlation. Clearly, the support of dimensionally-reduced data is an important analysis

case for scatterplots, but how to select between these possible strategies for scatterplot design is unclear, especially with the increased scale and complexity of datasets.

Designing for cognition and perception

There are yet other challenges faced by work that tackle how to design for data complexity in scatterplots. Central to many of these techniques is preserving the meaning of distance between objects as an indicator of similarity. In geography, the "first law of cartography" that states that objects closer in distance tend to be more similar [MacEachren, 1995], which has been adapted and codified to point spatializations (a.k.a, scatterplots) by Montello et al. [2003]. In particular, work has concentrated on the aspect ratio of the plot area, which can affect judgments of distance between objects Cleveland et al. [1988], Heer and Agrawala [2006], Talbot et al. [2011], as well as global judgments of correlation [Li et al., 2008, Rensink and Baldridge, 2010]. Scatterplots have also been a canonical player in testing perceptual issues of different visual encodings, including probing just-noticeable differences in point lightness [Li et al., 2010b], point size [Li et al., 2010a], comprehension of group statistics between point classes [Gleicher et al., 2013], and the judgment of linear correlation [Li et al., 2008]. Though the scope of these works concentrate on specific design decisions, combining these strategies can help to derive effective scatterplot design.

4.2.2 Typologies and Taxonomies

Typologies and taxonomies use abstraction to extract similarities and differences between concepts without unnecessary dependence on the particulars of individual implementations, as discussed earlier in this dissertation. A primary consideration common in many information visualization taxonomies is task, abstracting how a viewer interacts with and obtains information from a visualization (see Munzner [2014] for a high-level overview). Task is typically viewed on a continuum from high- to low-level [Rind et al., 2016]: a high-level task comprises an analysis goal [Brehmer and Munzner, 2013, Schulz et al., 2013], while a low-level task captures the exact information that viewers pull out of a visual representation [Casner, 1991, Healey et al., 1996] or describes bite-sized analyses [Amar et al.,

2005]. Another consideration may be understanding how factors and characteristics of the data can have ramifications on the visualization, as discussed by both Mackinlay [1986] and Ellis and Dix [2007]. These taxonomies, along with many others, help to standardize the lexicon and assist in the incremental progress toward tailoring effective design for a given analysis goal.

These taxonomies discuss visualizations in a general case, making it difficult to apply these organizations to influence the designs of specific visualizations in practice—though exceptions exist: Sedlmair et al.'s taxonomy for dimensionality reduction [2013], Sedlmair et al.'s taxonomy of cluster separation factors [2012b], and Lee et al.'s taxonomy for graph data [2006]. By focusing on the single scatterplot case, the goal in this paper is to create a framework with an impact statement similar to the design space goal set out in Schulz et al. [2013]. A framework should consolidate many similar but disparately presented research under a single lens to drive the framework forward, by explicitly examining the trade-offs between different strategies. By organizing research in this way, such a framework would concretize (or, in the words of Schulz et al., externalize) implicit design decisions to explicitly organize how designs work—helping to teach practitioners and researchers, clarifying design requirements, and making good abstractions that have practical value. This also has the advantage of identifying open areas for future research by identifying voids in the design space—as an example, it may become clear through the organization that a strategy does not exist for a particular set of factors. Therefore, our goal throughout this work is to create a framework specific for scatterplot-like designs, helping both practitioners and tool-builders to choose the correct design, given both the analysis goal and the characteristics of their data.

4.3 Scatterplot Tasks

While many task taxonomies have been constructed for general information visualization or even for specific data types (e.g., graphs [Lee et al., 2006]), we are not aware of such a task analysis specific for scatterplots. With a coverage of the space of the analysis tasks that concern scatterplots, a task list can allow for discrimination between designs—helping to

identify why one design may work better in a particular analysis scenario over another. We seek to identify tasks that form the building blocks for all analysis done with scatterplots, covering both low-level and high-level tasks. Such a list should be data domain-agnostic, which would allow for creating actionable abstractions of specialized scatterplot-like designs for specific data domains in similar analysis scenarios.

To formulate the seeds for this task list, we collected model tasks from a variety of sources in the data visualization literature, including papers performing empirical evaluation [Rensink and Baldridge, 2010, Tory et al., 2007], picking "good" views of correlation and clustering [Li et al., 2008, 2010b, Sips et al., 2009, Tatu et al., 2010], design studies of analyst scenarios [Brehmer et al., 2014a, Sedlmair et al., 2013], technique papers [Bertini and Santucci, 2006, Cottam et al., 2013, Tuan Nhon Dang et al., 2010], and even position papers [Correll and Gleicher, 2012]. The list of 23 collected model tasks, their source, and common categories are detailed in the supplemental material.

To abstract these model tasks, we asked four data visualization researchers (two faculty, two senior doctoral students; 5–10 years of experience) to perform a card-sort and group tasks together based on their similarity (see Spencer [2009] for an introduction). This card-sort strategy has precedent in the visualization community—such as Roth's application for deriving a set of cartographical interaction intents [2013a]. We asked them to use an open card-sort (no predefined categories titles, nor prescribed number of categories), and arrived with several categorizations of tasks. With minor disagreement, exploratory and cluster analysis generated consensus groups consisted of tasks that we then labeled *open-ended browsing and exploring*, *cluster rationalization*, *density judgments*, *dimension rationalization*, *multi-scatterplot tasks*, and *trend analysis*. Due to our concentration on single plot designs, we discarded the *multi-scatterplot tasks* category.

With these seed categories, we refined these categories post-hoc to generate a complete picture of the space (Table 4.1). We refocused the *trend analysis* category from the card-sort to *explore neighborhood* (#5), which captures obtaining aggregate statistics about a group [Cleveland, 1985, Gleicher et al., 2013, Healey et al., 1996], or identifying the similarities and differences among objects in a spatial region [Tory et al., 2007]. To expand the *browsing and exploring* category into representative tasks, we use the taxonomy from Casner [1991]

	#	Task	Description
object-centric	1	Identify object	Identify the referent from the representation
	2	Locate object	Find a particular object in its new spatialization
	3	Verify object	Reconcile attribute of an object with its spatialization (or other encoding)
	4	Object comparison	Do objects have similar attributes? Are these objects similar in some way?
browsing	5	Explore neighborhood	Explore the properties of objects in a neighborhood
	6	Search for known motif	Find a particular known pattern (cluster, correlation)
	7	Explore data	Look for things that look unusual, global trends
aggregate-level	8	Characterize distribution	Do objects cluster? Part of a manifold? Range of values?
	9	Identify anomalies	Find objects that do not match the 'modal' distribution
	10	Identify correlation	Determine level of correlation
	11	Numerosity comparison	Compare the numerosity/density in different regions of the graph
	12	Understand distances	Understanding a given spatialization (e.g., relative distances)

Table 4.1: Our list of abstracted analysis tasks that are performed with scatterplots: model tasks gathered from the literature, categorized with a card sort, and refined through reconcilation with visualization taxonomies.

to separate directed and undirected search, yielding the two tasks *search for known motif* (#6) and *explore data* (#7).

Judgments of distribution are another common task—while many papers concentrate on finding clusters [Brehmer et al., 2014a, Sips et al., 2009, Tatu et al., 2010], identifying other distributions such as manifolds are also important in many analysis scenarios [Li et al., 2010b, Sedlmair et al., 2013]—giving rise to the *characterize distribution* task (#8). From the *browsing* category, we also explicitly partitioned the task of *identifying anomalies* (distributional-specific outliers, #9) due to the common trade-off of scatterplot designs utilizing aggregation [Elmqvist and Fekete, 2010]. *Identifying correlation* (#10) between the two dimensions in a scatterplot is a canonical task with scatterplots [Cleveland, 1985, Friendly and Denis, 2005], which has prompted empirical studies of how correlation is identified in scatterplots [Li et al., 2008, Rensink and Baldridge, 2010]. We adapt the derived *density judgment* category to *numerosity comparison* (#11), which captures tasks that coarsely compare the numbers of objects embedded in spatial regions within the scatterplot. The last task in our list is *understand distances* (#12), capturing elements of the derived *dimension rationalization* category, to capture tasks of using and judging distances as a metric space against an object-embedded subspace [Montello et al., 2003, Sedlmair et al., 2013].

Many high-level tasks have been captured through this refinement process—dealing with sets of objects and understanding trends, distributions, and numerosity. We augment the derived tasks from the card-sort also to capture single-cardinality, object-centric tasks, such as look-up and *identifying* an object's spatialization (#1), searching for and *locating* an object in a scatterplot (#2), and *verifying* an object's spatialization within the plot (#3). As a pair to exploring the neighborhood (#5), *object comparison* (#4) involves comparing the visually-mapped (and non-mapped) attributes of a pair of objects to determine the relationship or similarity between the two data items [Cleveland, 1985]. These operations represent low-level operations in the visualization literature, stemming from Casner's analysis taxonomy [Casner, 1991] and repeated in others [Brehmer and Munzner, 2013, Munzner, 2014, Schulz et al., 2013].

This process results in twelve abstracted tasks (Table 4.1) that we use to help frame our discussion throughout this chapter. This collection of tasks is the first derived collection

of tasks that are specific to scatterplots, and abstracts the range of tasks from a variety of scatterplot designs. We note that these tasks may vary in terms of levels of abstraction or cognitive complexity, but represent distinct parts of an viewer analysis scenario. These tasks represent individual, separate intents and operations on behalf of a viewer to derive information from a scatterplot.

A complex analysis task performed within an analysis scenario, such as correlation discovery in high-dimensional data, may involve several of these tasks, used as building blocks, to achieve an analysis goal, similar to the task construction presented in other task taxonomies [Brehmer and Munzner, 2013, Schulz et al., 2013]. As an example, consider one of the model tasks collected: "match clusters and classes" (from Brehmer et al. [2014a]). This analysis goal can be composed of tasks #6, #5, and #4: search for known motif (find clusters), explore neighborhood (inspect objects within the cluster), and object comparison (inspect class membership of objects).

The curation of this list allows us to focus on supporting these tasks downstream in gauging and evaluating the task performance on scatterplots, based on data characteristics and design strategies. The task list helps to cover the range of tasks done with scatterplots over a wide range of analysis scenarios, but it is not able to capture how data characteristics may make some designs intractable. Nonetheless, we can use these tasks as a factor to distinguish the strengths and weaknesses of scatterplot-like designs, and we can provide better coverage of analysis scenarios when paired with data characteristics.

4.4 Data Characteristics

Many characteristics of the data (such as data size and distribution) may influence the design of an appropriate scatterplot. Similar to capturing the tasks of scatterplots, collecting, abstracting, and connecting relevant data characteristics will allow a more complete characterization of task effectiveness in the space of scatterplot designs. Here, we survey the challenges in particular sets of data characteristics, and discuss designs to support these characteristics in the design decisions (§4.5) and linking (§4.6) sections. The data attributes that we consider in this work are summarized in Table 4.2, along with reference articles.

Data Attribute	Possible Values	Relevant Work
Class label	No class label, 2-4 classes, 5+ classes	Elliott and Rensink [2015], Gramazio et al. [2014], Sips et al. [2009]
Num. of points	Small (<10), medium (10–100), large (100–1000), very large (>1000)	Cottam et al. [2013], Gleicher et al. [2013], Keim et al. [2010], Mayorga and Gleicher [2013], Tory et al. [2007]
Num. of dimensions	Two continuous, two derived, or >2 dimensions	Best et al. [2006], Chan et al. [2010], Sedlmair et al. [2013]
Spatial nature	Dimensions do/do not map to spatial position	MacEachren [1995], Montello et al. [2003]
Data distribution	Random, linear correlation, overlap, manifolds, clusters	Bertini et al. [2011], Li et al. [2008], Rensink and Baldridge [2010], Sedlmair et al. [2013], Sips et al. [2009], Tatu et al. [2010], Dang and Wilkinson [2014], Wilkinson et al. [2005]

Table 4.2: The data attributes considered in our work, with work inspiring these distinctions. The number of points are quantized into bins based on their overdraw effect on design decisions—numbers given are relevant for the 400×400 plot and 6×6 mark sizes shown in this chapter [Urribarri and Castro, 2017].

There has been precedence of capturing relevant data characteristics in scatterplots, both explicitly and implicitly. Implicit representations (that is, designs that do not encode summary or representative statistics) develop responsive designs to varying data characteristics, such as encodings that scale to support increased numbers of points (see Sedlmair et al. [2012b] for relevant factors in cluster separation). Explicit representations use quantitative metrics to capture different features of data characteristics. For example, the re-introduction of scagnostics by Wilkinson et al. [2005] to the information visualization community allows for metric calculation of very particular distribution characteristics. Capturing these relevant characteristics can help to quickly whittle down the space of applicable techniques when considering a large combination of dimensions or a large space of scatterplot-like designs.

A common data characteristic that prompts design consideration is an increased number of objects to represent. A critical threshold in understandability is reached when the number of objects to visualize approaches the limit of available screen-space to show individual points. Very clearly, the *number of points* to consider in a scatterplot will affect the appropriateness of a design, dependent on the screen-space available for the plot (see Urribarri and Castro [2017] for a discussion)—though the number of points may not affect some analysis tasks [Gleicher et al., 2013]. We quantize this factor into bins, where the bins prompt different design strategies to handle issues of data scale, such as the issue of overdraw. The data scale for the bins are dependent on the mark size and the plot size [Urribarri and Castro, 2017]—for example, larger plots with have higher thresholds for "large" numbers of points. Viewers can pick out individual marks and their referents at *small* numbers, while it is more difficult to pick out individual points at a *medium* scale. A *large* number of marks starts to exhibit problems of overdraw, while a *very large* number of points can only be displayed in aggregate. At larger data scales, designs tend to make use of aggregation to handle the data scale (§4.5).

Related to the number of objects, multiple data series are often shown in the same plot to compare distributions between and among groups [Sips et al., 2009]. A *class label* identifies different data series by discriminating points by shape or color. These labels can allow tasks to be performed on series in aggregate, which may or may not cause issues of distraction when performing tasks on individual series [Elliott and Rensink, 2015, Gleicher et al., 2013, Gramazio et al., 2014]. We discuss some relevant designs in the discussion (§4.7), though we do not consider multi-variate encodings (such as glyphs) in this chapter. In addition, although some designs map dimensions of data to encodings of the marks themselves (such as mapping a continuous dimension to the mark size, known as a bubble chart **?**), we do not consider supporting additional data beyond the two positional dimensions in this chapter.

While we concentrate on two-dimensional scatterplots in this chapter, the *number of dimensions* of the data under consideration is also an important consideration to make. We make a distinction between visualizing objects with two continuous dimensions [Best et al., 2006], two derived dimensions (from a process such as principle-components analysis

Sedlmair et al. [2013]), and visualizing a subset of dimensions from objects with more than two continuous dimensions (such as considering a high-dimensional system Chan et al. [2010]), as these scenarios effect design choices. Depending on the dimensionality considered, some tasks such as understanding correlation and distribution of the data may need additional design scaffolding, requiring viewer interaction to understand correlation throughout the dataset. As another example, clustering takes on different meanings depending on whether the data is projected from some high-dimensional space or being positioned based on two attributes (e.g., absolute object similarity vs. subspace).

We can also consider dimensions that *do* and *do not map to spatial position* [Montello et al., 2003]; this distinction can affect how viewers interpret distance as object similarity in the plot. This is particularly important in cartography, where fidelity of individual points can be a critical factor to maintain. Additionally, depending on the analysis task, aggregation to irregular boundaries of semantic value (such as borders of counties) are likely with spatial data. Non-spatial data does not necessarily place paramount importance on distances implying similarity between objects.

Lastly, the expected *distribution of the data* can affect the performance of various tasks—points can cluster, form distinct correlations, or can even stack. Scagnostics [Wilkinson et al., 2005] provides a list of nine types of relationships between two continuous variables. We seek a more focused list of relationship categories that designs may target. These five categories are not necessarily exclusive to one another, but serve to separate how task appropriateness may be affected by the distribution of the data. Data that groups into **clusters** (*clumpy* in scagnostics) is an area of interest in the literature [Sips et al., 2009], as identifying why clusters occur can be an analysis task. The distribution of the data can also group into semantically-meaningful shapes such as **manifolds** (*coherence* in scagnostics), which can be relevant for other analyst tasks [Brehmer et al., 2014a, Sedlmair et al., 2013].

The potential for overdraw increases if the distribution of the data involves points that have very **similar dimensional values**, and scagnostics captures this sentiment with *clumpiness*. Data that contains a **linear correlation** (*trend* in scagnostics) are critical to communicate effectively, and much research in both the statistics and information visualization communities has focused on good design decisions to emphasize potential correlation,

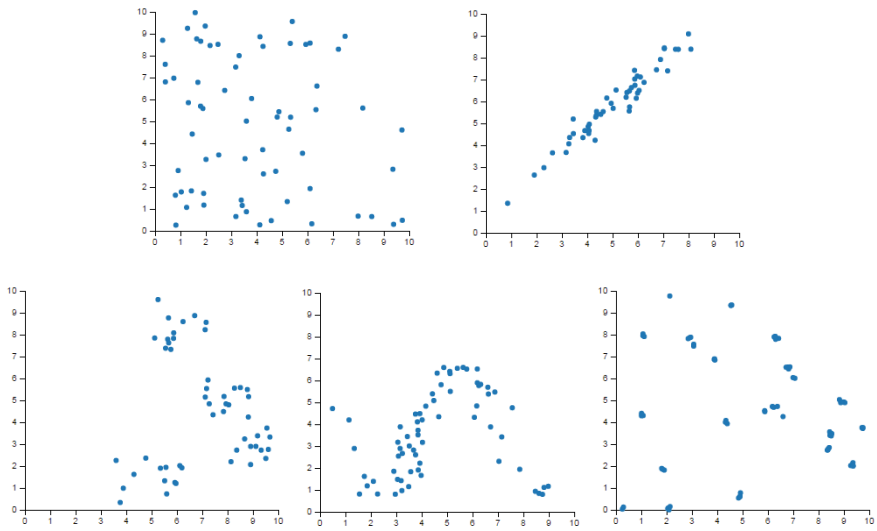


Figure 4.2: Sample distributions captured with the five types of data distributions considered: randomly distributed, linear correlation, clustering, manifold (matching a discernable function), and overlapping points.

including picking the ideal aspect ratio for the plot [Fink et al., 2013, Rensink, 2017] and adding visual embellishments such as a trendline. Lastly, data that appears randomly distributed (where a discernable trend is hard to determine) is an important case to consider, and has the potential to confound several potential analyst tasks.

There are also a variety of visual design choices that can be made to enhance viewer's understanding of the data and provide scaffolding for particular analysis tasks. These data attributes specify potential challenges in representing the data, which prompts particular sets of design decisions.

4.5 Design Decisions

To understand the breadth of the space of scatterplot designs, we collected designs and organized a taxonomy of design decisions for scatterplots (see Table 4.3). We posit that any scatterplot-like visualization will use some combination of these design variables in its construction. We identify these design variables through a separate literature survey (disconnected from §4.3). By enumerating these design decisions, we can use the previously-listed factors of analysis task and data characteristics to help determine the applicability of design decisions.

We collect design decisions from their use in visualization research papers. These decisions range in complexity from simple design decisions of the points (their color, size, and texture) to more advanced grouping techniques (convex hull shapes, KDE blending). By identifying these design decisions, we can start to identify strengths of different design strategies while also providing a framework in which to organize future techniques according to their task support. Combined with the list of scatterplot analysis tasks, the full framework (§4.3–4.5) can be used to link design variables with their task support, conditioned on the given characteristics of the data (see application in §4.6). We provide the full details about these sources, their design decisions, and which tasks and data characteristics are supported by each strategy in the published manuscript [Sarikaya and Gleicher, 2018].

Relevant manuscripts were gathered through a keyword search methodology. We searched the titles and abstracts of articles published in the Information Visualization journal proceedings, EuroVis proceedings, Pacific Vis proceedings, and all VIS proceedings (SciVis/Vis, InfoVis, VAST) from 2009 to 2017 (3040 papers) for any instances of the string "scatter." Our query returned 117 results, of which 62 were relevant to scatterplots (a common matching element was "scattering," a component of rendering). We then perused these articles, pulling out information such as the anticipated support of scatterplot-specific tasks, the design strategy utilized, and the types of encodings evaluated or explicitly supported in the presented technique or experiment. This information is available in the material associated with Sarikaya and Gleicher [2018].

A benefit of building this space is that it articulates the range of scatterplots that different decisions make. This space thereby suggests potential programmer and designer abstractions that should support this range. To assist in realizing scaterplot designs in practice, we have developed a D3-based [Bostock et al., 2011] library for scatterplot-like designs, called *d3-twodim* and discussed in Chapter 7. The library allows programmers to experiment with designs utilizing both SVG and WebGL, adding automatic interaction support for linked components such as dropdown menus and tooltips.

4.5.1 Clustering of Design Choices

After collecting design choices (right-most column of Table 4.3), we group these choices together into clusters. Grouping these choices together helps to clarify the purpose and application for appropriate design. In the clustering, these decisions modify the design of the marks themselves (*point encoding*), group points by a visual technique (*point grouping*), modify the marks' position (*point position*), or add annotations, call-outs, or other amenities to the scatterplot (*graph amenities*). These clusters are discussed below, with some discussion of strategies that utilize these design decisions to support a particular analysis goal.

Point Encodings cover the design variables that can be applied to marks to represent objects in the graph, and can serve to differentiate encoded objects from one another. These types of common encodings can be considered as the decisions to be made on "marks," as many visualization grammars describe them (such as graphics grammar of Wilkinson [2005]). Examples of these encodings are color, size, shape, and orientation. Careful use of these encodings can take advantage of pre-attentive processing of the human visual system [Cleveland, 1985, Ware, 2012], directing the viewer's attention to particular subsets or patterns. Combinations of encodings can help viewers select subsets of points relevant to their exploration. Deliberate use of these encodings to group points together can be considered as an implicit grouping, which we discuss next.

Point Grouping decisions serve either to simplify the visual product by aggregating similar items together or to differentiate items from one another. Their role tends to further constrain and focus the overall message of the visualization. The term "grouping" is analogous to the usage of the term *abstraction* in visualization (see the use in Elmqvist and Fekete [2010]). As we use it here, however, we consider grouping to be a superset of aggregation design decisions, and the choice of design strategy will emphasize a particular message. Design decisions under the point grouping designation drive task performance by narrowing the scope of potential insights—for example, collecting points into bins sacrifices the fidelity of item detail but exposes and highlights distributions of data.

Our framework organizes point grouping into *implicit* and *explicit* groupings. Implicit

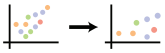
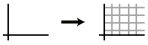
Cluster	Design Choice	Example
Point Encoding	Color	
	Size	
	Symbols	
	Outline	
	Opacity	
	Texture	
	Depth of Field	
	Blurriness	
Point Grouping	Representation Type	
	Positional Binning	
	Polygon Enclosure	
	Shape Abstraction	
Point Position	Subsampling	
	Displacement	
	Animation	
	Projection	
	Zooming	
Graph Amenities	Grid Lines	
	Axis Ticks	
	Legend	
	Trend Lines	
	Annotations	

Table 4.3: A categorization of design decisions available to the scatterplot designer, which are clustered into four categories. Each of these categories can be used to gauge appropriate design strategies.

grouping uses *point encodings* to identify points as belonging to similar groups by categorization, distance in attribute value, or other similarity metric. Implicit strategies show data as points and rely on the viewer's perception to group points together, generally by way of gestalt grouping [Ware, 2012, Ziemkiewicz and Kosara, 2010]. In contrast, explicit grouping reduces object-specific fidelity to abstract marks and communicate aggregate, high-level judgments of the data. A canonical example in this space is binning [Carr et al., 1987], where group statistics of marks are collected for small, regular spatial regions—while this trades off the fidelity of individual marks for more aggregate judgments, it may be better able to communicate the numerosity differences in different regions in the plot.

Polygon enclosure, by continuous, majority, or full convex hull means, can simplify areas of high visual noise to indicate highly dense regions. For example, VisIRR [Choo et al., 2015] uses a simplified ellipse to collect groups of points together while also redundantly encoding category with color. In particular, these strategies can be composited with point position strategies to explicitly support a small set of analysis tasks, similar to the strategy of Splatterplots [Mayorga and Gleicher, 2013]—group marks together, then explicitly restore and style individual marks that fall outside the grouped region. These enclosures need not be enclosed shapes; Cleveland and McGill [1984b] use smoothings (upper/lower residuals) to emphasize the correlation of the two axes. Shape abstraction can also be used to emphasize the trend of a distribution, such as used by Yates et al. [2014] to emphasize different logical implications between the two axes of a scatterplot.

Point Position — Scatterplots tend to display data items by creating a spatialization by two continuous attributes. However, some designs modify point position to pack more information into the visualization (e.g., reducing dimensionality) or emphasize particular areas of the graph (zooming and displacement). These decisions are made to emphasize support for particular tasks over absolute accuracy and faith to the original data. By modifying point position, these strategies combat issues of overdraw stemming from either a poor distribution of data or simply coping with the inevitable overdraw with too many objects for the screen-space. Utilizing these strategies can help to emphasize distributional judgments, assist in identifying and tracking objects of interest, reduce more than two

dimensions to a familiar scatterplot design, and visually organize data for subsequent decisions (both point encoding and grouping strategies). As an example, Chen et al. [2014] use "smart" subsampling to convey distributions of multiple series while minimizing overdraw of individual marks. Keim et al. [2010] use a subspace warp to effectively use unneeded screen-space to emphasize distribution judgments.

Graph Amenities — Annotations and other scaffolding can help the viewer navigate a scatterplot. Examples of these strategies include grid lines, axis ticks, object labeling, encoding legends, and trendlines. These amenities help by orienting the viewer (e.g., axis ticks) and providing additional information (e.g., legends, sensitivity lines) relevant to analysis. Much like point position strategies, scaffolds of this type can serve to emphasize the particular message of the visualization, specifically helping viewers to complete object-centric tasks. Most critically, these amenities can help a viewer navigate the visualization by highlighting relevant items through annotation, provide distributional context with tick lines, and highlight potential correlations with trend lines.

4.5.2 Interaction Intents

Interaction commonly accompanies scatterplots to support the intentions of viewers. While interactions are not necessarily *visual* design decisions, they are commonly used in conjunction with visual strategies to support the tasks of the viewer. Brehmer and Munzner [2013] also motivate the inclusion of interaction intents as the "how" in their task typology—a critical component to support changing the visual strategy to support viewers in their analysis. In the same vein as Amar et al. [2005], these intents signify the desire of the viewer to change the granularity of the visualization or change the reference frame. These intents indicate a desire to directly contrast or evolve the current set of design decisions with a new set, incorporating the strategies that will make the appropriate design variable changes to support the given intent. With a change in the design, the spectrum of task support changes—potentially in a deliberate way.

Interaction can signal that the viewer wants to change to a view that is more appropriate for their desired analysis task. For example, a viewer may want to *focus analysis on relevant*

items, in which case they may be able to interact with items in the visualization (direct interaction), brushing (selection), or by interacting with a linked component (e.g., text-box, table of attributes). To emphasize the relevant objects or groups, a *point encoding* could be assigned to highlight the relevant marks. Two common pivots deal with changing the level of granularity—seeing more detail or less detail; "elaborating" and "summarizing" by the taxonomy of Schulz et al. [2013]. *Seeing more detail* could involve actions such as zooming or jittering, both examples of *point position* design strategies. *Seeing less detail* could involve abstraction through subsampling or aggregation, examples of *point grouping* strategies.

Thinking about interaction as an intent to change the visual design to support a competing task can help rationalize the controls that a viewer has. For example, InterAxis [Kim et al., 2016] allows viewers to use exemplar objects to dynamically weight and re-project the dataset to identify related objects, allowing viewers to change their frame of reference. Many lensing techniques, such as MoleView by Hurter et al. [2011], use the lens to select relevant types of items as a way of reducing distraction from other overlapping elements. MoleView also supports aggregation behavior (such as edge bundling) within the lens to further reduce element complexity. This highlights an intent from the viewer to switch from a high-level overview of the data toward a more localized, detailed neighborhood exploration setting. These intents provide another layer of abstraction to group design decisions for supporting analysis tasks.

4.6 Using the Framework to Drive Design

Our framework suggests that scatterplot designs should be matched to the tasks and data characteristics that they are designed to support. The tasks and data characteristics form a high-dimensional space—any scenario is a point in this space. For any one point in the space, we can determine which design decisions are appropriate. Creating a map of this entire space is challenging because it is large. Even if we divide the axes into discrete buckets (such as §4.4), we are left with 12 (tasks) $\times$ 4 (points) $\times$ 3 (dims) $\times$ 2 (spatial) $\times$ 5 (distribution), yielding a grid of over 4300 discrete scatterplot scenarios.

For each scenario, we seek to determine which of the five design cluster strategies are

appropriate. In some cases, this will be easy to determine. There may be examples that prove the effectiveness of a design strategy for a scenario, or reasoning about factors can determine inappropriateness (e.g., point encodings are inappropriate for identifying an object among millions of points). In other cases, however, the decision may not be so clear: it may require an empirical study to determine if a design is effective for a scenario; there is the potential for a specific novel design that effectively employs the strategy for a scenario; or the strategy is only effective under certain circumstances.

The massive grid of effectiveness decisions would be attractive, but also infeasible to fully realize because of its size. Additionally, many entries would only be our current subjective assessment subject to change based on newly discovered designs or empirical evidence. For these reasons, we have not attempted to provide the full table. Instead, we have given our (current, subjective) assessments for a large portion of the grid, provide a web-based tool for exploring this high-dimensional table[1], and show a representative "slice" of the table below, showing how our framework can be used to match scatterplot designs to analysis scenarios.

While a pre-determined table of appropriateness would be convenient, our framework can be applied without it. The important part of the framework is that it enumerates the factors to consider and the design choices–informing the structure of the grid. Specific assessments of appropriateness should be the subjective opinion of the designer based on the concerns detailed in Sections 4.3–4.5. Examples of applying this type of analysis for a range of scenarios is provided in the next section.

4.6.1 A Slice of the Space: Tasks and Design Strategies

We illustrate our framework with a small slice of the entire grid: a specific set of data characteristics, the entire range of tasks, and the entire set of design strategies. For the sake of demonstration of the framework and to support discussion of the current high-level trends and strategies in scatterplot design, we are providing 60 out of the 4300 cells of the overall table. To demonstrate an interesting reference point where the design of a faceless scatterplot becomes intractable for many tasks, we choose a particular set of

[1] `http://graphics.cs.wisc.edu/Vis/scattertasks`

data characteristics. This slice *fixes the set of data characteristics* to a moderate number of objects and number of classes, in an *unstructured* distribution of scattered data. We note that we could also take an alternative slice of the map, with 10 points, no class label, in a random distribution, and the map would provide a wildly different set of appropriateness measures.

With this map and aforementioned slice in particular, we examine how and why certain encoding decisions can or cannot support particular analysis tasks. As an example, identifying and comparing numerosity in a faceless scatterplot can start to become challenging when many points are overlapping, masking the viewer's determination of density (and thereby suggesting a design change). In Table 4.4, we denote appropriate design decisions with a ✔, potential design support with ✔*, support possible with an accompanying design decision with ✧, and inappropriate support with ✘. These determinations are made and motivated by our assessments of the state-of-the-art, existence proofs of design and interaction techniques in the research literature (informed by our survey detailed in §4.5), and empirical experimentation of encoding decisions for specific viewer tasks. In the prose below, we describe specific decisions in the slice and describe their extrapolation to the broader table. We also contrast suggested designs with designs that may work better in other scenarios with different data characteristics.

At a high-level, appropriateness for design decisions for various tasks begins to expose clusters of similarly-supported tasks. The cells in the slice are referenced by the task (a number) and the encoding type (a representing letter). We discuss some of the shortcomings of typical strategies for scatterplot design, and provide pointers to exemplar systems that can scaffold the desired analysis tasks.

- *Difficult to support aggregate-level tasks with point encodings* (**9A–11B**) — Tasks 9, 10, and 11 deal with aggregate-level tasks that seek to uncover characteristics about the data on a global scale, either by identifying those marks that are outliers or anomalies, gauging correlation across the dataset, or understanding object density across the graph area. Due to the aggregate nature of these tasks, utilizing the strategy of how marks are encoded (**A**) or moving point positions (**B**) will not help. The similarity of encoding

	Task	A Point encoding	B Point position	C Point grouping	D Interaction intent	E Graph amenities
1	Identify object	✔	✔	✧	✔	✔*
2	Locate object	✔	✧	✧	✔	✔
3	Verify object	✔	✔*	✧	✔	✔
4	Compare objects	✔	✔	✧	✔	✔
5	Explore neighborhood	✔	✔	✔	✔	✔
6	Search for motif	✔	✔	✔	✔	✔*
7	Explore data	✔	✔	✔	✔	✔
8	Characterize distribution	✔	✔	✔	✧	✔
9	Find anomalies	✧	✔*	✧	✔*	✔
10	Identify correlation	✘	✘	✔	✘	✔
11	Characterize numerosity	✘	✘	✔	✘	✘
12	Characterize distances	✔*	✔	✔*	✔*	✔

Table 4.4: A 2D slice of the task support map by clusterings of visual encodings, with data characteristics set to a "large" number of points with a few number of classes in a non-clustered position (so the possibility of overdraw exists). ✔ denotes general support, ✔* denotes support in particular situations (discussed in prose), ✧ requires concurrent support from other encodings, while ✘ identifies no improvement to task support.

strategy effectiveness among these three tasks suggest that it may be fruitful consider these three tasks under an "aggregate-level" umbrella, where encoding decisions made to support these tasks stand in opposition to "object-level" tasks. In scenarios with fewer points, it may be possible to support these tasks with implicit grouping. However, such approaches would not apply in situations with significant overdraw.

- *Unclear how to design interaction and amenities for aggregate-level tasks* (**10D, 11D–E**) — There is a clear gap in designing interactions (**D**) for aggregate-level tasks such as identifying correlation (#10) or performing comparisons in object numerosity (#11). Direct manipulation approaches have been proposed [Saket et al., 2017], though exactly how to

prompt viewers to interact with the visualization to promote correlation or numerical understanding is unclear. While graph amenities (**E**) can help to see correlation (such as overlaying a trend line over the data), identifying and comparing numerosity in multiple areas on the plot becomes difficult with many annotations and call-outs. To potentially address these issues, landscape views [Tory et al., 2007] use point grouping strategies to emphasize numerosity judgments.

- *Losing mark fidelity with **point grouping*** (**1C–4C, 9C, 12C**) — Point grouping (**C**) provides a way to abstract and convey a particular narrative about the data. By aggregating marks into large visual shapes, designs using point grouping strategies lose the support of object-centric tasks such as finding outliers and comparing objects. As an example, performing continuous aggregation via KDE [Scott, 2008] would support judgments of comparing numerosity across the plot (**C11**), but would not support object-centric tasks such as *locate object* (**C2**).

  However, by compositing aggregation operations with point encodings, point positions, and interaction intents, object-centric tasks can be supported. As an example, an interaction where a viewer hovers over a filled-in region could subsequently highlight exemplar points, which could then be explicitly selected for object comparison (#4). Many scatterplot-like techniques use a composition to restore support for object-centric tasks, such as Splatterplots [Mayorga and Gleicher, 2013] and the sampling strategy by Chen et al. [2014]. Exactly what design patterns that may prompt a viewer or an analyst to engage with an aggregated display to perform an object-centric task remains an open question, though many systems use interactions such as brushing to populate an external component, such as a "selected" list.

For specific concerns in Table 4.4, there exist several classes of design strategies that can help bolster the efficacy of analytical tasks.

- *Supporting distance judgments* (**12A–D**) — The distance between marks (task #12) takes on a different meaning based on the dimensionality being visualized. The marks may be placed based on two continuous attributes of the objects, where the distance between marks communicates the distance in attribute space, or the marks could be placed based

on two dimensionally-reduced, derived dimensions, where data is placed based on the total similarity of its continuous attributes. To support the analysis of dimensionally-reduced data in a scatterplot, many visual analytics systems provide scaffolding by amenities or external linked components. Dis-Function [Brown et al., 2012], for example, supports direct interaction of individual marks to update the similarity projection of the entire high-dimensional dataset.

- *Dealing with overdraw* (**1E, 6E, 9D**) — With significant data, the possibility of overdraw or masking of object-representing marks exists. This hurts detection of individual points, and designs have been constructed to preserve judgments of numerosity [Cottam et al., 2013, Keim et al., 2010, Mayorga and Gleicher, 2013, Tory et al., 2007] or use alternative methods such as visual aggregation (§2.3.2, Gleicher et al. [2013]) to preserve statistical judgments. Many of these designs do not use graph amenities (**1E**). However, paired with a lensing technique (see generally Tominski et al. [2016]), this analysis scenario could be supported. Similarly, exposing a given distributional motif (**6E**) is difficult given that this motif may not be known to the visualization designer *a priori*—but specialized amenity techniques such as drawing moment lines [Chan et al., 2010] can convey an aggregate sense of a motif. With increased numbers of points, however, these amenities can themselves exacerbate the problem of overdraw.

  Again in an overdraw scenario, it may be difficult to distinguish outliers or anomalies with an interaction intent (**9D**)—how might an analyst specify "show me the outliers" directly within the plot? One strategy is to compose strategies with other operations: Splatterplots [Mayorga and Gleicher, 2013] explicitly selects those marks that fall outside thresholded density regions, and ensures those marks are visible while zooming the plot.

- *Consciously supporting object-centric tasks* (**1C–4C, 9C, 2B**) — Marks that represent objects are needed to obtain information about individual objects. Object-specific tasks (#1–3) and object-centric tasks (#4, 9), such as compare objects, depend on the specific marks for a viewer to perform their desired analysis, but many point grouping techniques (**C**) aggregate marks together. To be able to support these tasks, several different types of

strategies have been developed; a common strategy to support these object-centric tasks is to provide a external filtering component that selects objects based on semantic content or viewer-defined thresholds, then highlights the selected objects as marks overlaying the aggregate encodings. This strategy can also help finding the positions of marks if the points are moved (**2B**). Many interactive lensing techniques have also been developed, where a viewer can mouse-over to see more detail of the objects contained within the lens scope [Heimerl et al., 2016, Tominski et al., 2016].

The supplemental material [Sarikaya and Gleicher, 2018] provides a listing of 62 strategies that handle the analysis scenarios raised within this linking table, organized by the characteristics of data supported, the analysis tasks supported, and the types of design decisions used. We illustrate common themes in scatterplot design in the discussion section.

4.7 Discussion

Throughout this chapter, we have developed a framework to discuss the design of scatterplots. Using the task list (§4.3), we are able to focus our attention on how those tasks are supported by scatterplot designs and affected by characteristics of the data. Trends of task support by data characteristics for traditional scatterplots have been identified, and lead to suggestions of design strategies to support the desired tasks. These suggestions lead to trade-offs in the design of scatterplots. There are instances in scatterplot design where the circumstances of the data prevent a single design strategy from supporting all tasks. For example, a density-based encoding with thousands of points can support the task of numerosity comparison easily, but needs conscious design support for identifying outliers.

The following themes highlight potential challenges in designing effective scatterplots, and suggest strategies for supporting common analysis scenarios.

Visual Complexity / Too Many Points — Dealing with visual clutter has been the focus of many visualization techniques and taxonomies. In particular, Ellis and Dix [2007] explore a wide range of strategies and the trade-offs between them. Many of the techniques that

we found through our literature search employed some method of visual simplification, explicitly supporting some analysis tasks while weakening support for others. These strategies generally fall under the categories of point grouping and point position strategies. Point grouping strategies generally abstract groups of points into fewer distinct visual structures, emphasizing numerosity and distributional judgments at the expense of tasks dealing with individual objects. Through the point grouping process, however, the ability to identify both outliers and anomalies usually becomes hindered (aggregate-level tasks).

On the other hand, point position strategies such as projection and animation can pack additional structural information into a scatterplot without sacrificing the viewer's ability to execute element-specific tasks. While these methods necessarily modify the "true state" of each mark's spatialization, these methods can emphasize hidden or overlapping structure based on the characteristics of the data. As an example, generalized scatter plots [Keim et al., 2010] warp the subspace of the plot area to maximize the use of space (point position) and utilize a KDE-like point grouping strategy to emphasize the numerosity of points.

A common problem in scatterplots is the problem of overdraw when there are simply too many marks for the available chart area. Similar to the visual complexity problem, both grouping and position strategies can help alleviate the issues of incomprehensibility at scale. A generalized set of point grouping strategies provide different levels of support for analysis tasks. In principle, the plan of what features of the data to communicate determines the scope of design strategies that emphasize those characteristics.

Demonstrating *distributions* is well-supported by density-driven encodings, such as shape binning [Carr et al., 1987] or continuous density estimation [Scott, 2008]. By abstracting away individual point marks and using visual weight to communicate relative numerosity, we can support the aggregate-level tasks such as characterize distribution or identify correlation at the expense of object-centric tasks such as object comparison or verify object. While examples of these density-driven encodings are numerous, there are particular design details within these strategies that have trade-offs of support between the scatterplot analysis tasks.

Effectively communicating *numerosity* can often be concurrently supported by strategies that emphasize distribution, though caveats exist. For strategies that support kernel den-

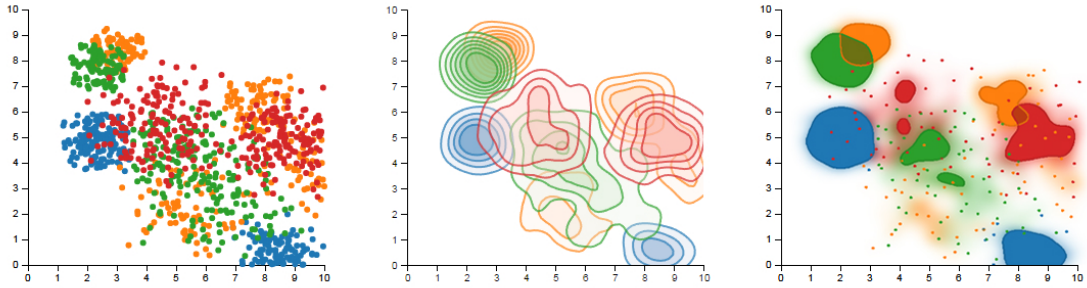


Figure 4.3: Three different designs (left-to-right: traditional scatterplot, contour map [Collins et al., 2009], and Splatterplot [Mayorga and Gleicher, 2013]) display different information about the same 100 item, four class (mapped to color) dataset. While the traditional scatterplot exhibits some overdraw, the two alternative approaches use point grouping techniques to emphasize numerosity and distribution comparison tasks. The contour map conveys density gradients, while the Splatterplot uses thresholded regions to convey dense areas.

sity estimation [Keim et al., 2010, Mayorga and Gleicher, 2013], a thresholded region may communicate the range of a high-number of points, but without a complex contour map [Collins et al., 2009], it can be difficult to compare approximate number of points. Aggregation commonly has computational complexity on the order of the number of points, though some (such as Splatterplots) may use the GPU to compute repetitive density estimation. Computationally simpler strategies can utilize blur [Staib et al., 2016] or alpha encodings [Cottam et al., 2013] to communicate relative numerosity of marks, given an appropriate normalization dependent on the current view [Matejka et al., 2015].

Figure 4.3 shows a side-by-side comparison of three scatterplot designs, all displaying the same dataset with a "medium" number of points—individual points can be discerned, and class distribution is still apparent in a faceless scatterplot. However, not all tasks are equally supported by each design—the faceless scatterplot supports object-centric tasks (#1–3) with some overdraw, while colored contour maps Collins et al. [2009] (center) eschew object-centric tasks to focus attention on distributions and densities. Comparatively, the Splatterplot Mayorga and Gleicher [2013] (right) shows outlier points, but aggregates points together using a thresholded KDE, providing a sense of locality of dense regions between the classes. While both the contour map and Splatterplot use point grouping strategies, the contour map provides more information about density information than the

Splatterplot—which could sway a designer's choice of design strategy depending on the analysis goals of the viewer.

Differentiating Groups of Marks / Too Many Classes — Many strategies have been proposed to differentiate groups of marks. Much early work has concentrated on the perceptual grouping of points, with Cleveland [1985] mentioning ways of emphasizing groups of points by using distinct encodings. Mackinlay [1986] provides a perceptual ordering of encoding decisions, Ware [2012] describes the perceptual basis behind the ordering of the visual variables, and Li et al. explores perceptual sensitivity to these factors in scatterplot applications [2008, 2010a, 2010b]. Using point encodings to separate marks into groups is a very common trait, usually to split data into separate series or categories. While supporting object-centric tasks such as *locate object* and *identify anomalies*, these type of solutions also promote the exploration of data by creating interesting structures in the data to peruse.

An open problem in scatterplot design is how to communicate large numbers of series or categorization for marks. In many analysis scenarios, the number of classes to consider may number from the tens to hundreds of classifications, where comparison in numerosity or distribution between any number of series may be important to the analysis. A core limiting factor is the number of encodings to use to distinguish marks from each other: color has a fidelity of around 12 distinct hues [Ware, 2012], which rapidly declines with smaller visual area [Stone et al., 2014]. Different shapes can also provide additional separation, but again suffer at small sizes. Some strategies allow the viewer to focus on a small subset of series and place all other data into a "background" group [Kincaid and Dejgaard, 2009, Staib et al., 2016], or take advantage of hierarchy within the data to group similar objects together [Elmqvist and Fekete, 2010]. The literature lacks techniques for handling large numbers of classes, even though the problem is common, often appearing in humanities analysis contexts [Alexander and Gleicher, 2016, Heimerl et al., 2016].

Communicating High-Level Statistics — In many scenarios, it may be advantageous to communicate the distribution of the data or highlight potential correlation. Studies such as those by Gleicher et al. [2013] have shown how encoding decisions can affect viewer judgments of group statistics without explicit representation by graph amenities or point

grouping (such as the smoothings as presented by Cleveland and McGill [1984b]). While it may be important to explicitly support statistics of the data through graph amenities (e.g. annotations or showing a confidence interval), supporting statistical judgments implicitly can help in analyses where the specific statistics important for analyses are not known *a priori*. Some designs use shape aggregation to emphasize distributions, such as pictograms by Lehmann et al. [2015] or glyph SPLOMs by Yates et al. [2014], sacrificing object-level judgments for rapid distribution judgments.

Too Many Dimensions — Pragmatically, the number of dimensions should not affect the appearance of a scatterplot, as only two dimensions are shown. However, tasks performed with dimensionally-reduced or projected data tend to differ from the tasks done on two-dimensional data. To this end, many dimensionally-reduced scenarios contain extra detail about objects and can permit direct manipulation to feed back into the dimension-reduction algorithm. Strategies such as Dis-Function [Brown et al., 2012] or InterAxis [Kim et al., 2016] use direct viewer interaction to drive the semantic clustering of similar objects together. To support visualizing multiple dimensions without precomputation, multi-axis embeddings such as star coordinates [Kandogan, 2001] or their orthographic variant [Lehmann and Theisel, 2013] can expose clusters in a two-dimensional embedding. Many of these scenarios concentrate on object-centric and distributional scenarios that highlight the semantic similarity between objects.

4.8 Conclusion

Scatterplots are a visualization design widely applicable to a large range of analysis scenarios. With the many different design strategies available to select from, understanding the trade-offs between the many design choices is challenging. In this chapter, we have introduced a framework to help determine the design appropriateness for task support, and show how this framework can help gauge task performance that is dependent on characteristics of the data. With the characterization of this design space, we have described the challenges, existing solutions for these challenges, and potential areas for innovation in scatterplot design.

With both the table slice presented in Section 4.6 and the gradient of task support by changing pairs of data characteristics, we build an understanding of how appropriate scatterplot designs are dependent on the factors of task and data. Although we capture how these factors address design, we acknowledge that the process for determining the tasks that a viewer will perform with a scatterplot is difficult to determine automatically, and generally falls under the purview of a designer as part of an iterative design process [Sedlmair et al., 2012a].

The scatterplot tasks described herein, however, greatly reduces the space of scenarios to support by clustering support together. Designing to support object-centric, aggregate-level, and outlier tasks seem to be at odds with one another. Designs should therefore deliberately support one set of tasks over another, using interaction to support viewers in obtaining information that is difficult to obtain with a design strategy. This framework provides a model and an organization to help visualization designers choose and rationalize their design decisions for scatterplots, based on the factors of data and task.

Interaction to support the viewer's exploration and understanding of the data is explored in the next two chapters. Both chapters describe a biologically-motivated use case of exploring large amounts of data, utilizing both summarization techniques and interaction to recover both low-level, object-specific information and high-level trends of their respective datasets.

5 CASE STUDY: VISUALIZING VALIDATION FOR PROTEIN SURFACE CLASSIFIERS

In the first of the case studies, we explore the visualization space of validating machine learning methods. In this particular scenario, the visualization treats the classifier as a black box, and visualizes the performance of the classifier over a test set of proteins. In this particular case, since the classifier makes decisions of a three-dimensional protein structure binding a particular ligand, a visualization that shows performance necessarily needs to be three-dimensional. However, practitioners commonly have to understand and evaluate classifier performance over a large set of proteins, in the tens or hundreds of proteins. Summary statistics (e.g., "classifier was 90% accurate") leaves out important insights, such as consistent spatial or feature motifs where the classifier fails to recognize a valid binding site. Visualizing the performance of the classifier on the protein can highlight issues in the classifier's decision making process and can suggest avenues for training iteration of the classifier. Simply showing 200 proteins is infeasible in 3D—by the nature of occlusion (data on the protein opposite of the camera view), information would be left out of an overview.

In this chapter, we explore the ramifications and trade-offs of providing a two-dimensional overview of inherently three-dimensional data, with design and development process similar to Sedlmair et al. [2012a]. With the reduction in dimensionality from a three-dimensional to a two-dimensional representation, some information must be given up—either spatial fidelity or aggregate trends. We highlight the trade-offs of different summary representations (§5.3.2), and show how these different summary representations can be used to explore the data in the classifier case studies (§5.5).

To be able to summarize structural data into a two-dimensional summary, the data is projected into a lower subspace through many techniques, such as linearizing the decisions into a heatmap, using quilted blocks to maintain proportions of decisions per molecule, or capturing three-dimensional cohesion through a squarified treemap. As each decision carries only one of four values (true/false positive, true/false negative), and a protein is made up of many of these decisions, the chief information that these summaries need to convey is trends and outliers, followed by frequency and distribution. The design

of the overview allows a viewer to choose between summary representations, and we assert in this chapter that these different designs elicit different types of information about classifier performance. More detail about these decisions are in Section 5.3. A publication representing the work in this chapter was presented at the 2014 EuroVis conference [Sarikaya et al., 2014].

5.1 Overview

The core challenge of structural biology is to understand how the form of a molecule connects to its function. A key approach is the development of computational models that predict locations on the surfaces of molecules where, for example, the molecule will bind with another. Such models are validated by comparing their results with experimentally-derived ground truth. Inspecting these results on a single molecule is challenging as the similarities and differences are spread around a 3D surface that has occlusions and irregular shape. Detailed examination of the results of an experiment involving dozens of molecules is prohibitive. Bioinformaticians typically resort to examining only aggregate statistics, losing the opportunity to examine the details of the experiments to find interesting cases within the set or to provide feedback to the modeling process.

This chapter introduces an approach to explore the results of classification validation experiments. We focus on surface classification, where the model predicts whether each location on a protein's surface is likely to bind to another molecule. The challenge is to provide an overview of the results of an entire validation experiment with many molecules, allowing the viewer to identify locations of interest, while retaining facilities for examining the specific details of interesting sites. Our approach addresses this challenge with a small-multiples view designed to allow a viewer to see aggregate properties on individual molecules as well as to identify details of interest that lead to these properties. This overview is connected to a detail view that provides specialized navigation controls over the 3D structures, allowing regions of interest to be examined rapidly.

The approach is based on two key ideas. The first is that an overview can be designed specifically for understanding aggregate properties over multiple scales. Using 3D views of

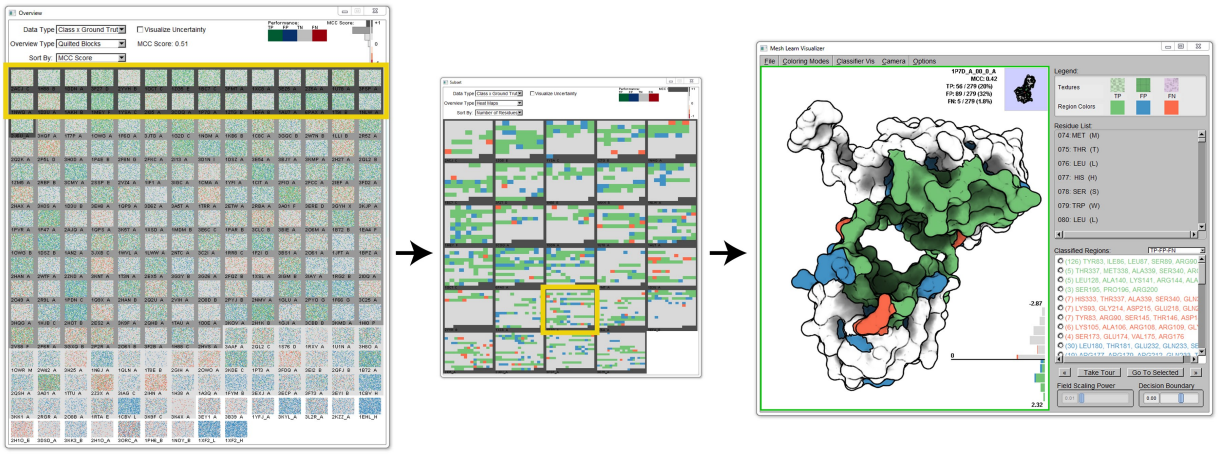


Figure 5.1: Visualization of a validation experiment for a DNA-binding surface classifier. The corpus overview (left) is configured to display each molecule as a quilted glyph and orders these glyphs by classifier performance to show how performance varies over the molecules. Selected molecules (left, yellow box) are visualized as heatmaps in a subset view (middle) and ordered by molecule size to help localize the positions of errors relative to correct answers. The detail view (right) shows a selected molecule to confirm that most errors (blue, red) are close to the correctly found binding site (green).

molecules for the overview is impractical, as they require more space, more time to navigate each surface, and do not afford quick summarization. Instead, we build on recent work demonstrating that people can perceive aggregate properties over certain kinds of displays to design 2D views that allow the viewer to quickly assess classifier results across an entire set of molecules. This overview can be used to identify specific molecules to explore more closely in 3D, as well as to suggest features of interest on these surfaces. The second key idea is to use information about the viewer's interest to drive navigation along the surface. Our approach abstracts information over the surface to identify discrete regions of interest, which are used to create navigation controls aligned with the information in the overview.

Bioinformatics classifier experiments are common: for example, a recent survey [Irsoy et al., 2012] notes several hundred papers per year, in just three bioinformatics journals, involve presenting classifier validation results. The survey notes that most of these papers report only simple statistics, at best providing statistical confidence tests. Better tools for exploring the results of these experiments could improve predictive model development and application. For example, identifying specific molecules or classes of molecules where a classifier performs well may help in understanding the generality of the predictive

model. Identifying false positives may help in selecting challenging decoys. Patterns of false negatives may suggest alternative mechanisms not represented or captured in the model training process. Individual errors can be assessed to see if they are near misses or anomalies.

The results of classifier validation experiments have a simple form. For each object in a corpus, every location has a prediction (positive or negative) marked by its correctness (true or false). This work specifically considers protein surface classifiers, where the objects are protein surfaces and the locations are 3D positions along those surfaces. However, the problem of comprehending validation experiments exists in other bioinformatics domains, for example in classifying properties of sequences. The ideas of our approach should apply more generally. Although the detail views are specific to 3D shapes, similar ones could be developed to navigate long sequences.

In providing a system that addresses the needs of scientists assessing the results of surface classifier experiments, our work makes several contributions. We demonstrate that recent results in how viewers perceive visual information in aggregate can inform overview designs and provide examples showing how glyph designs can be created to support a variety of aggregate assessment tasks. We also show how region grouping can be applied to provide interface support for exploration tasks. While our approach is demonstrated in a specific application for examining molecules, we believe that the contributions generalize to similar domains.

To present our approach, we begin by considering related work in the visualization of molecules and machine learning results. We then discuss our overview display, exploring a space of designs that leverage perceptual principles to support various assessment tasks. Next, we describe our detail view, explaining our specialized molecular view and data-driven interaction designs that aggregate regions of potential interest to support the viewer's tasks. Finally, we conclude by describing a prototype implementation and example use cases.

5.2 Background

The design of overview displays for large data collections is an important topic in visualization, see Hornbæk and Hertzum [2011] for a survey of the issues and approaches. To be effective, overviews must be designed to support efficient visual processing by considering the abilities of the perceptual system, see Ware [2012] for more detail.

Perceptual science has built an understanding of the types of visual features that can be processed efficiently. The visualization community has built upon this knowledge to guide display design (see Healey and Enns [2012] for a survey). These perceptually efficient, or "pre-attentive," features allow for rapid search in complex displays by, for example, leveraging "pop-out" effects, where properly encoded features can be located quickly in a large complex displays. Our design follows these guidelines, using salient colors that allow the viewer to rapidly find important locations.

Recent research in perceptual science demonstrates that people can efficiently estimate aggregate properties of large collections of objects. For example, people can estimate numerosity [Halberda et al., 2006] and average size [Ariely, 2001]. Recent work in visualization (*cf.* Correll et al. [2012], Albers et al. [2014]) shows that this applies to visualization displays, enabling *visual aggregation* where the viewer estimates statistical properties. Certain types of visual features, such as color, can be averaged more effectively than others [Correll et al., 2012], and performance can be further improved through other design choices. Albers et al. [2014] consider a range of estimation tasks and show how different visual designs can lead to displays that excel at different tasks. Our approach follows previous examples of visualization systems specifically designed with these principles in mind (e.g., [Albers et al., 2011, Correll et al., 2012]).

Flexible views can be effective to highlight patterns of interest when those patterns are not known *a priori.* A common approach for creating flexible views is to use a small-multiples display [Tufte, 1990]. The ability to reorder juxtaposed small-multiples can help adapt them to support different tasks by spatially clustering objects with related properties. While the basic concept of a reorderable display was introduced by Bertin [1983], work by Slingsby et al. [2009] has highlighted the power of reordering to support answering

the range of questions a viewer may seek. Our overview applies this flexible reordering approach.

5.2.1 Molecular Visualization

Many existing visualization tools have been developed to support molecular visualization tasks (see O'Donoghue et al. [2010] for a survey). Modern molecular graphics systems provide many different views of large molecules, including views that encode data fields on molecular surfaces. Such programs can be used to show results of classifier experiments on specific molecules; however, they are not tailored to the specific needs of understanding classifier performance across a corpus of molecules. Our approach provides a similar view, but augments it with interaction techniques specific to the task, coupling it with an overview display.

A handful of existing systems provide visualization over collections of molecules. Some systems, such as the web interface to the Protein Data Bank (PDB) [Berman et al., 2000], provide visual galleries using standard 3D displays as icons for molecules. Karve and Gleicher [2007] demonstrate a system designed to provide an overview of the metadata of a collection of proteins, but the design does not consider specific tasks or support classification experiments, and their glyphs are not optimized for pre-attentive summarization. Khazanov and Carlson [2013] present statistical properties over a large collection of molecules, but use only standard summary statistic visualizations such as bar and line charts, and provide no connections to specific molecules. To the best of our knowledge, our approach is the first to consider providing an overview of a collection of molecules that supports both summarization and detail finding.

5.2.2 Machine Learning Visualization

Visualization for machine learning applications strives to communicate either the internals of the predictive process or trends in the outputs. Tools for understanding prediction processes are tailored to particular machine learning algorithms, such as linear SVMs [Caragea et al., 2001], decision trees [van den Elzen and van Wijk, 2011], and hidden

Markov Models [Dai and Cheng, 2008]. Our work falls into the latter, helping viewers to understand results.

Summarizing the results of a classifier can be problematic as there are different types of errors in a model [Witten et al., 2011]. Several methods of quantifying performance exist [Powers, 2011]. Basic metrics such as accuracy, precision, and recall do not capture the error profile and are problematic for biased distributions. The Matthews correlation coefficient (MCC) [Matthews, 1975] accounts for class distribution to compare a classifier's performance to chance, but still provides only a single summary statistic for performance.

Visual methods provide a more detailed presentation of machine learning results. Talbot et al. [2009] use an interactive visualization to let the user explore the contributions of individual models in an ensemble scenario. Fails and Olsen [2003] show interactive adjustment of parameters to tune a predictive model. The user can explore shortcomings in the model and make adjustments to improve it. Our work also provides this type of feedback.

5.3 Experiment Overviews

Experimental results for binary classifiers consist of a large number of classification decisions, each of which has one of four outcomes (true positive (TP), false positive (FP), true negative (TN) and false negative (FN)), that form the binary confusion matrix [Stehman, 1997]. While the data is simple, it grows quickly: experiments generally are run over dozens of molecules, and there are tens to hundreds of decisions for each molecule.

Our goal is to provide an overview of the collection of decisions and corresponding experimental results. In addition to showing overall performance, the overview should help identify the specific molecules, and even parts of molecules, for which the classifier performs well or not. For instance, it should allow the viewer to assess whether performance is uniform across all molecules or varying; to identify groups of molecules that perform similarly; to identify outliers or anomalies that may represent problems; or to see high-level patterns of performance between molecules. These assessments can occur at different scales, for example an anomaly might be a particular molecule whose performance skews

results, or a family of molecules skewed by concentrated groups of false negatives.

Our approach uses two main ideas to support this range of needs. First, it emphasizes flexibility, allowing the viewer to reconfigure the display to suit their task. It allows for re-ordering and for selecting among a set of glyph types. Second, the glyph designs are designed to support rapid visual aggregation. This allows the viewer to see both the aggregate properties of the data and low-level details that form these aggregates.

5.3.1 Reorderable Small-Multiples Design

The overview uses a small-multiples display, where each molecule is shown as a small glyph in a grid. Different designs for the glyphs are provided (described below), but they share features that allow for pre-attentive summarization. Each glyph relies heavily on color encodings. Color supports pop-out [Healey and Enns, 2012] and pre-attentive summarization [Albers et al., 2014], making it useful for conveying aggregate properties as well as highlighting outliers. Each glyph has a gray border whose lightness gives an indication of the overall performance (MCC score, with darker borders representing a higher value).

The small multiples can be reordered to explore different types of questions. For instance, ordering by performance (e.g., accuracy or MCC) places molecules with similar performance together and allows for rapidly identifying strong and weak performers. Ordering by molecule name facilitates finding a specific item of interest. Ordering by metadata (properties of each molecule) emphasizes correlations between that property and performance. Coupling the different orderings with different glyph designs provides a wide range of configurations to support various questions. For example, sorting by the size of the molecule and choosing an appropriate glyph type can not only show whether large molecules perform better or worse than others, but can also indicate whether the errors form large groups on the molecules.

The overview provides some basic interaction features that directly support common tasks. Selecting a glyph can open the molecule in the detail view for closer examination. Sets of glyphs can be selected and opened in a new overview window, allowing for more

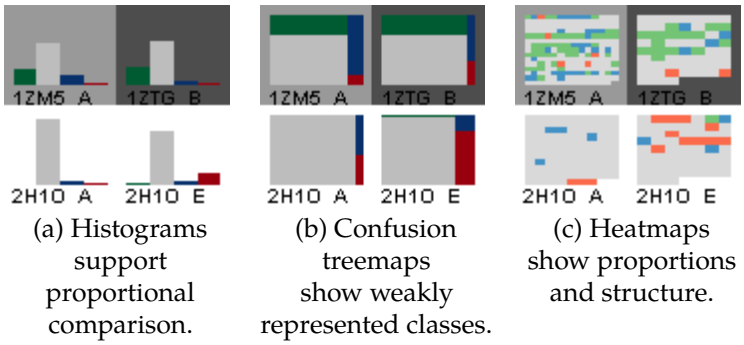
localized analysis of subsets of the dataset. The user can annotate the glyphs in order to track which molecules have already been examined or should be explored in greater detail.

5.3.2 Glyph Design

3D views of the molecule would be difficult to see in the small space of the glyphs. Additionally, because at least half of the molecule is occluded, some form of navigation or surface unfolding would be required to make an assessment of the whole surface. The highly irregular shapes of molecules, with their significant pockets and protrusions, make meaningful flattening difficult. Our current set of glyphs does not provide a 3D or flattened view and therefore generally does not convey the spatial layout of data on the molecule.

Instead, we leverage nonspatial 2D views that sacrifice information about the spatial arrangement of elements in order to remedy occlusion problems inherent in 3D views. Further, these views can be designed to support rapid visual comparisons both within an element and between multiple elements by leveraging visual variables in the encoding design. In our system, we leverage color as the dominant channel to encode classification decisions to support rapid visual assessment, mapping TP to green, FP to blue, FN to red, and TN to gray. This color mapping leverages salience to support classifier analysis tasks by considering *a priori* characteristics of the data and task — TN are common and are mapped to gray to decrease their saliency, while FN represent highly undesirable classifications that generally require attention and are mapped to red.

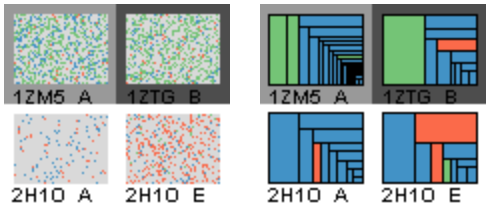
Our system allows the user to switch between different glyph designs in order to configure the display to their task. Each design supports certain kinds of visual queries.
Histograms (Figure 5.2a) are a standard encoding and are useful for showing the performance distribution within a specific molecule. However, they become harder to interpret when a single class dominates, and do not afford efficient visual aggregation.
Confusion Matrix Treemaps (Figure 5.2b) sacrifice some of the inter-class fidelity of histograms, but better show weakly represented classes and make better use of space to afford pre-attentive area judgements between elements. A vertical divider delineates the proportion of correct classifications (left side), and incorrect classifications (right), providing a quick indication of the predictive accuracy.



(a) Histograms support proportional comparison.

(b) Confusion treemaps show weakly represented classes.

(c) Heatmaps show proportions and structure.



(d) Quilted blocks show summary statistics pre-attentively.

(e) Cluster plots show spatial cluster sizes.

Figure 5.2: Different glyph encodings for overviews afford different observations about the data.

Heatmaps (Figure 5.2c) encode the data from each decision using small patches visualized in sequence order. Because the size of the patches in a glyph is inversely proportional to the number of decisions in the corresponding molecule, this display gives a sense of the molecule's size. Averaging and proportion estimation is supported by the color encoded design. As residue sequence order is related to spatial proximity, this view can also provide some insight into how the various points are grouped along the surface.

Quilted Blocks (Figure 5.2d) are similar to heatmaps, except that the placement of the pixels from each color patch is randomized within the glyph. This representation sacrifices any sense of the structure of the sample to make pre-attentive summary statistics easier to access [Correll et al., 2012] and to help highlight performance patterns at the molecular level.

Cluster Plots (Figure 5.2e) use a squarified treemap representation [Bruls et al., 2000] to indicate groups of similar classes that are spatially clustered on the surface. While the glyph does not convey the positions of the groups, it does convey their number and size.

The overview can visualize either raw binary decisions (positive or negative) or supplement these decisions with the respective confidence of each decision. The viewer can optionally show confidence values in the heatmap and quilted displays. When visualizing confidence data, each of the four colors is replaced by a three-step sequential color ramp in the same hue drawn from Colorbrewer [Harrower and Brewer, 2003].

5.4 Detail View

While 2D overviews trade-off spatial information to communicate performance across multiple proteins, showing classifier decisions in the context of the surface is important for understanding the connection between molecular shape, chemical properties, and the decisions. Unfortunately, presenting the classification results on a molecular surface has several problems. Because the 3D view necessarily occludes much of the surface, especially when there are pockets and crevices, finding locations of interest can be challenging. Also, when examining multiple disjoint features, the viewer must remember which ones have already been examined. Our approach attempts to remedy such issues for classification results presented on the molecular surface through interaction techniques designed to assist search and memory.

The detail view is a standard molecular surface visualization, with triangle mesh surfaces created using MSMS [Sanner et al., 1996]. Following Tarini et al. [2006], we apply stylized shading to convey shape, which includes ambient occlusion shading and contours. We perform visibility calculations for ambient occlusion on the bounding sphere about the surface. Predictions are encoded on the molecular surface using the same color scheme as the overview.

5.4.1 Regions of Clustered Data

Protein classification necessarily discretizes the molecular surface, though this sampling hides the fact that the molecular surface is a continuous field. The viewer's perceptual system can group similar points to identify patches [Palmer, 1992]; however, when the

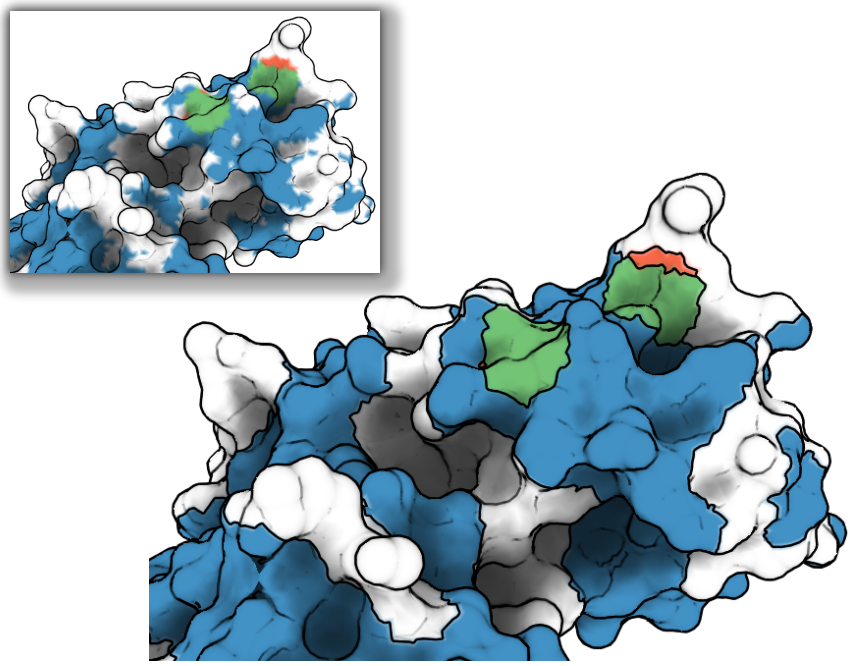


Figure 5.3: Clustering similar values creates discrete regions that can be identified visually and by interaction.

sampled predictions alone do not form coherent visual structures due to issues such as noise or undersampling, perceptual grouping may be insufficient.

We simplify the extraction of high-level continuous patterns from classifier data by explicitly grouping predictions along the surface. This approach represents a trade-off of fidelity for simplicity: we sacrifice information about individual points in order to better characterize the high-level continuous properties of the surface. This surface grouping is illustrated in Figure 5.3. Classification regions are built by performing connected components on labeled vertices. The resulting boundaries are jagged, but precise. The regions can be simplified by smoothing region boundaries by the morphological operations of dilation and erosion [Serra, 1982].

Grouping points into clusters provides a number of benefits. Visually, it allows the display to emphasize the differences between groups by clearly marking the boundaries. Simplifying boundaries reduces visual noise, making high-level patterns more apparent. The resulting reduced set of elements also simplifies user interface support for interfacing with task-driven interaction techniques. For instance, the discrete list of clusters provides a visual checklist for the viewer to record regions they have already examined (Figure 5.5).

Coupling this list with automatic navigation, we eliminate the need to manually locate regions of interest along the surface. Such identification is particularly valuable in locating small regions.

5.4.2 Automatic Viewpoint Selection

Locating individual clusters can be challenging. While some clusters may be large and easily identifiable, others may be small, hidden in pockets, or occluded from the current viewpoint. Automatic viewpoint selection brings a selected cluster to the center of the viewport without requiring the user to manually navigate the surface. A user may navigate by selecting a cluster from a reorderable list. Our method for viewpoint selection builds on previous literature on finding optimal viewpoint navigation [Vázquez et al., 2001].

We characterize a "good" viewpoint as one that maximizes the visible area of the cluster. To simplify the search for the best viewpoint for a given cluster, our approach restricts camera positions and paths to a bounding sphere about the surface. Our implementation computes the visibility for each vertex of the surface mesh, from a sampling of directions, when the surface is first loaded. This is used for illumination computations to create ambient occlusion shading and is also used for automatic viewpoint selection. To find the best viewpoint for a region, the sampling direction from which the most vertices of the requested region are visible is selected. The corresponding point on the bounding sphere of the molecule is chosen for the new viewpoint. The viewing direction (look-at point) is chosen as the center of the region.

Transitions to a selected camera position are created by spherically interpolating the viewpoint on the bounding sphere, and linearly interpolating the look-at point. These smooth transitions help the viewer remain oriented when they select a region to transition to. These transitions also serve as the building block for "automatic tours," where the system generates a list of regions and shows them to the viewer in sequence. Such tours are useful to give an impression of the entire surface of a molecule.

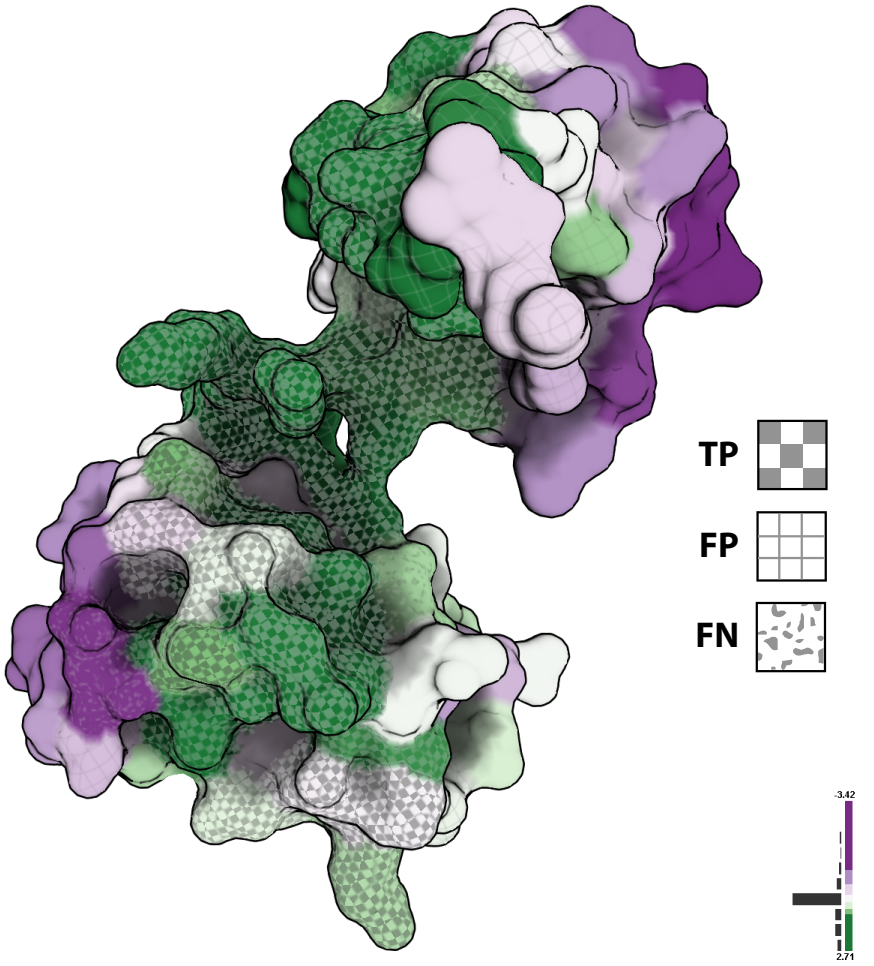


Figure 5.4: A multivariate encoding for a scalar field (shown as the purple-to-green color field) overlayed on classification values shown as procedural textures (checkerboard, grid, Perlin noise). Note how TP (checkerboard) and FP (grid) generally correlate with positive charge (green), suggesting a correlation between charge and positive predictions.

5.4.3 Predictions and Scalar Fields, Simultaneously

Molecular graphics programs frequently use surfaces to display scalar data fields such as electrostatic charge and hydrophobicity. Bivariate encodings can be used in order to make comparisons between these data fields and the classification decisions. Although bivariate color ramps can encode two fields [Ware, 2012], it is difficult to extract each independent dimension from the encoding [Ware, 2009]. Bivariate ramp design is further complicated by luminance changes introduced by shading on the molecular surfaces. Therefore, we instead use textures to convey the classifier decisions, and reserve color for encoding the

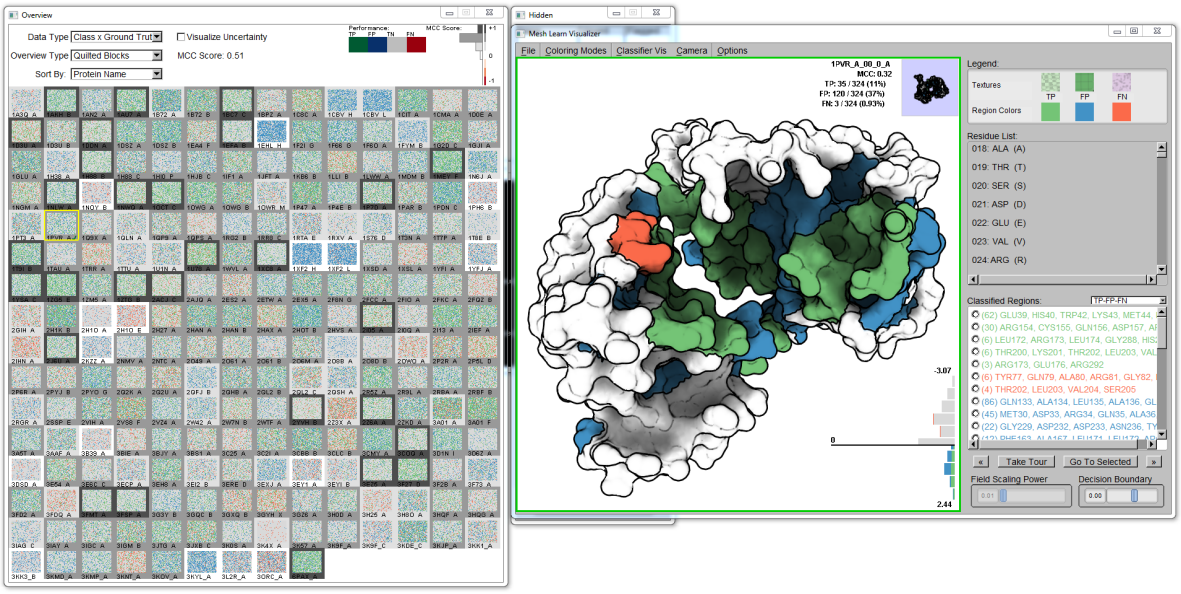


Figure 5.5: Our approach applied to the validation of a DNA-binding classifier. The overview window (left) displays the corpus rendered as quilted blocks (§5.3.2), giving an idea of aggregate performance across the corpus. The detail window (right) shows the clustered classifications (§5.4.1) for PDB: 1PVR_A, highlighted in yellow in the overview window. These clusters are itemized (lower right), allowing for highlighting regions of interest and automatic navigation to view a selected region.

field of interest.

Complex surfaces generally do not lend themselves well to traditional surface parameterization for two-dimensional texturing. We instead use 3D procedural textures [Perlin, 1985] as they can be mapped using only the coordinate system of the molecule. Classification results are depicted using three disparate textures (TP as checkers, FP as grid, FN as Perlin noise). For example, in Figure 5.4, the relationship between an input feature (electrostatic charge) and the classification result is visualized by encoding feature data with a seven-step, purple-to-green color ramp and classifications with texture. The scalar field color ramp is intentionally distinct from the colors used to encode classification results alone to avoid confusion. A histogram (bottom-right in Figure 5.4) displays the distribution of the scalar field feature alongside the boundaries of the color ramp. This graph serves as a control widget for updating the transfer function, allowing the color ramp to be modified interactively.

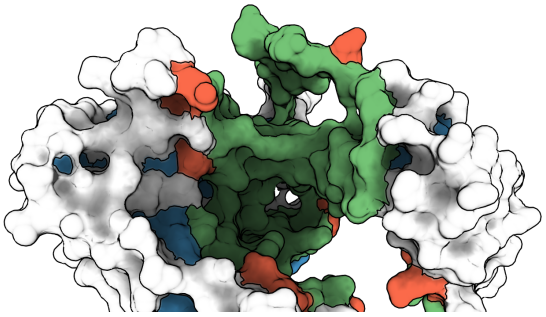
5.4.4 Dynamic Decision Boundary

To further help understand the classifier outputs, the decision boundary for the classifier can be adjusted in the detail view. Changing this boundary affects the classifications of predictions by raising or lowering the threshold of a positive prediction. The detail view (Figure 5.5) contains a histogram illustrating the distribution of classifications for the visualized molecule in the context of the current decision boundary. The viewer can directly manipulate this boundary to highlight predictions with a high confidence while reclassifying the remainder and can push the new decision boundary back to the overview to reclassify the entire corpus.

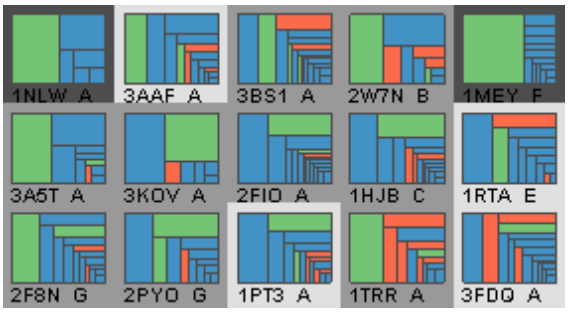
5.5 Use Cases

The prototype implementation of our approach is implemented in C++, using FLTK and OpenGL 3.3. The system can read in classification results for an entire corpus experiment in a few seconds. In all cases we have tried, the overviews are drawn in a fraction of a second so re-configurations of the overview display (reordering or changing glyph type) is nearly instantaneous. The surface meshes for each protein are generated using a standard external tool, MSMS Sanner et al. [1996]. While this tool may take up to several minutes to generate a surface mesh for a large protein, these meshes can be pre-computed before an interactive exploration. Our system can load a mesh and perform the visibility computations required for ambient occlusion and navigation in less than three seconds, even for very large molecules. Timing information used a machine with an Intel i7 920 (2.67 GHz) CPU and a nVidia GeForce GTS 250 graphics card.

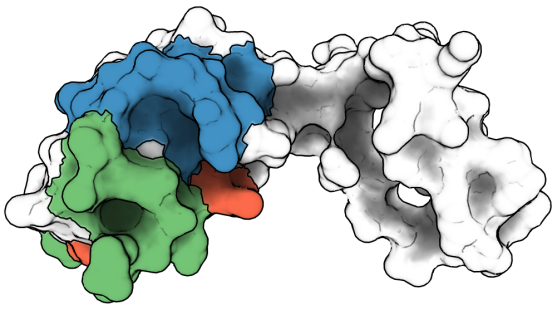
We demonstrate our methods using two protein classifier datasets: a DNA-binding classifier with a test corpus of 219 proteins (Figure 5.5) and a calcium-binding classifier with a test corpus of nine proteins (Figure 5.7). Prior to our tool, assessment of results was done by looking at tables of statistics, and by loading surface colors into standard molecular graphics tools. The executable and use cases are available online at the project website at `http://graphics.cs.wisc.edu/Vis/PSCVis/`.



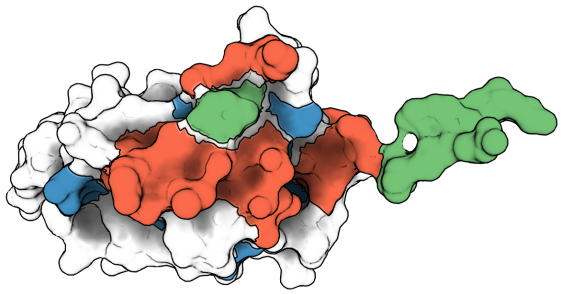
(a) PDB: 2I05_A, an example of good performance. A large pocket (green) holds DNA while FN and FP are on the fringes.



(b) The region cluster plot summary glyph enables identification of proteins having FP regions with similarly-sized TP regions.



(c) PDB: 2W7N_B, selected from the region cluster plot above, shows large region of FP adjacent to the discovered binding site.



(d) PDB: 3FDQ_A; the linear shape of the binding site leads to large regions of FN, suggesting alternative binding mechanisms.

Figure 5.6: Analyzing the spatial clustering of a DNA-binding classifier reveals high-level trends of classification.

5.5.1 DNA-binding Classifier

Figures 5.5 and 5.6 show a validation experiment of the DNA-binding, residue-granularity, predictive binding model named DNA-Binding Site Identifier (DBSI) [Zhu et al., 2013]. Ground truth labels indicate that DNA has been found to bind within five Angstroms of the residue in the crystallographic structure. The model performs well, in terms of summary statistics including F1 and MCC scores. However, closer examination of the validation results reveal more about its performance.

Figure 5.5 shows the DBSI test set (219 proteins, sizes of 41–932 residues) loaded into the visualization prototype. Using an overview with quilted blocks ordered by performance confirms the overall peformance, but shows three different types of errors. Molecules

with good overall performance (MCC) are predominantly TP, with some FN and FP. Mid-performing molecules often have some TP, but also large FP regions. Poorly performing molecules often have large amounts of FN.

To examine the first type of errors, a region cluster plot shows that many molecules have large TP regions, and many small incorrectly classified regions. Examining these clusters in a detail view (e.g., Figure 5.6a) shows that the small errors are usually at the fringes of a correctly identified site. Automated touring allows multiple examples to be examined rapidly to confirm this trend. These "near-misses" are unlikely to be meaningful in practice as precise localization is difficult because proteins are dynamic. However, it suggests that the classifier designers consider spatial grouping in order to improve their performance scores.

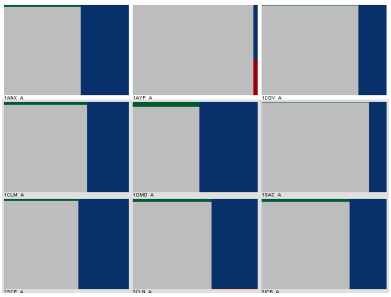
The region cluster overview also showed patterns in the larger errors. One trend was molecules with large regions of FP and TP (Figure 5.6b). The detailed views show the FP regions surrounded the TP regions (Figure 5.6c). Screenshots of the visualization were used to communicate results to scientists, who suggested explanations. For example, binding different sequences of DNA could result in minor conformational differences that change the label of nearby residues.

A third observation came from examination of some of the poor performing molecules. The overview identifies molecules with large false negative clusters. When examined in the detail view), they often have a false negative cluster with a long narrow shape (Figure 5.6d). The linear nature of the binding site does not seem to be captured by the classifier — instead of the typical conformation of the protein enveloping the DNA, the binding site of this particular protein seems to tuck itself into the grooves of DNA.

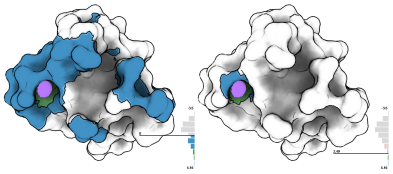
These three observations use elements of our approach, with chosen overviews leading to details. Each would have been difficult, or impossible, to make with the traditional approach of tables of statistics and manual inspection.

5.5.2 Calcium-binding Classifier

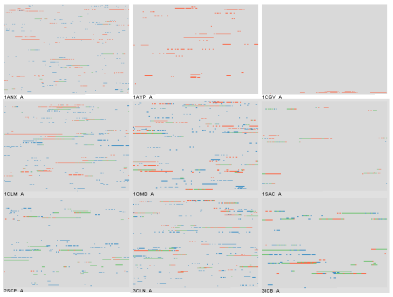
We applied our system to the validation of a calcium-binding classifier based on surface descriptors [Cipriano et al., 2012], but using a simpler machine learning approach than in



(a) Confusion treemaps shows the results of the calcium-binding classifier overestimate potential binding sites with significant FP.



(b) PDB: 3ICB_A showing classification regions with decision boundary 0.0 (left) and 2.61 (right).



(c) A heatmaps overview with the decision boundary changed to 2.61, showing fewer false positives, and the possible existence of multiple binding mechanisms.

Figure 5.7: Analysis of a surface descriptor-based, calcium-binding classifier. Modifying the decision boundary indicates that calcium may bind in multiple environments not adequately generalized by the classifier.

the paper. The validation experiment had 11 proteins. As decisions were made for each mesh vertex, each molecule had between 11k and 63k data points.

This classifier performs poorly over the test corpus (MCC: 0.163); this is shown in Figure 5.7a. The large number of FP (blue) shows that the classifier overestimates the number of binding sites. Examining a specific example (Figure 5.7b, left) confirms this trend. Adjusting the decision boundary to be more conservative (Figure 5.7b, right) better captures the true binding sites. Pushing the adjusted boundary to the entire test corpus reveals that the more conservative decision boundary causes entire binding sites to be missed (red, FN, Figure 5.7c).

Corpus-level visual inspection reveals some trends in the data and identifies the errors. The large number of small binding sites, and the diversity of these sites, suggests that calcium binds in many different kinds of environments. However, the error patterns show that while some sites are discovered correctly, many are missed. This suggests that the classifier is only capturing some of the ways that calcium may bind. The simple algorithm of the classifier, which cannot capture multiple modes, is insufficient; the complexity of the published algorithm may be warranted.

5.6 Discussion

This chapter introduces an approach for exploring protein surface classifier validation results. The approach couples an overview of a collection of molecules with a detail view for examining specific molecules. The overview helps not only to identify patterns of performance across the corpus, but also to find specific molecules of interest. The detail view is designed to address the search and memory issues involved with exploring complex objects.

There are several limitations to this work. At present, it does not support the comparison of multiple classifiers. While some limited support for adjusting the decision boundary is provided, we have no explicit mechanisms to compare the different patterns that occur from adjusting this boundary. We also do not provide any 2D summaries that convey the relative spatial layout of disjoint classifications. For example, none of the current encodings can show that the false classifications occur close to true ones. While the overview supports direct navigation to detailed views of specific molecules, it does not allow navigation to specific regions of interest within these molecules. While our approach should apply to classifiers for objects other than molecular surfaces, we have not tailored the system for such applications nor designed new detail views.

The flexibility of our overview is a tradeoff: the ability to reconfigure the display allows it to support a range of queries; however, this requires the user to make informed configuration choices. In time, we will evolve the set of options and provide guidance on how to match them with tasks. In practice, we believe that rapid reconfiguration allows a user to find an appropriate view, potentially discovering other perspectives on their data en route.

To date, the evaluation of our approach has been limited to a few anecdotes and use cases. While specific elements of our design could be evaluated in controlled studies, direct assessment of the overall approach is more challenging. Tests on controlled data sets can allow the confirmation that users can actually identify the kinds of performance patterns our system is designed to expose. However, a better validation of our approach will be its success at helping in the design of more effective classifiers. A challenge will

be to convince classifier developers of the potential value of close examination of their experimental results.

Even in our initial use cases, we have used the system to help reveal insights into the physical groupings of the classifications on protein surfaces. Summaries allowed identifying trends and selecting examples to explore in detail. The detail views enabled relating patterns of error to the performance of the classifiers.

6 CASE STUDY: VISUALIZING CO-OCCURRENCE IN POPULATIONS OF VIRAL GENOMES

In the second of the case studies, we look at the issues of identifying correlations between genomic mutations in viral populations. For this problem, we deal with uncertainty in what makes a "significant" correlation, as well as large amounts of data (upwards of 1GB uncompressed RNA sequence data). In order to support this problem with a visualization, we gathered requirements from virologist stakeholders and collaboratively developed a solution to address a scientific need. As with the previous case study, a successful design hinged on appropriately addressing the factors of data and viewer task. We present a negatively-received design in this chapter, and describe how this experience led to the final improved, more appropriate design of *CooccurViewer*.

To highlight significant co-occurrences of mutations in genomes, the summary visualizations used herein utilize filtering to identify and show only relevant genomic positions. Due to the ambiguity of what constitutes a significant co-occurrence, the thresholds for significance are viewer-controlled. Counts of mutated and non-mutated RNA bases are collected for each genomic position, and significant co-occurrences display these counts using a Sankey-like, parallel sets design technique. The main characteristics of data communicated through this visualization are correlation, though the distribution of co-occurrences through the genomic axis are also relevant for analysis (see Section 6.7). In order to emphasize correlations, signficant co-occurrences are filtered and aggregated to allow the viewer to quicly identify correlations that are relevant for analysis—focusing the analysis on this analysis scenario. A publication representing the work in this chapter was presented at the 2016 EuroVis conference [Sarikaya et al., 2016].

6.1 Overview

Many analytic activities involve understanding *events* in sequences. Events may be significant points in time-series data, locations in text documents, or positions along a genomic sequence. A wide variety of techniques in the visual analytics literature focus on identifying

and interpreting events as sparse sets of interesting locations in a sequence. However, the problem of identifying interesting patterns of *co-occurrence* of observations relating events together is much less studied. Examining co-occurrence requires considering a much larger space than with individual events: rather than the one-dimensional space of a sequence, co-occurrence must consider the space of all pairwise relationships. Additionally, analysis must consider incomplete data, as observations may not capture all pairs of events.

In this chapter, we present a design study for the problem of the identification and analysis of co-occurrences of mutations within DNA sequence data. In our design study we gather requirements, determine an abstraction of the problem, formulate a strategy based on prior research, evaluate prototypes, and arrive at a final visualization design, driven by participatory design with our collaborators. Through this process, we encountered issues of scale associated with displaying all potential correlations. A key idea in our strategy is to define metrics for quantifying "interestingness," affording a user-driven exploration of the space of correlations. While our motivating application is the population dynamics of viruses and correlation of mutations, we believe the lessons from this design study have broader applicability to discovering correlations in other one-dimensional sequence data.

The specific biological question we consider involves the mutation patterns that a virus makes over the course of its infection in a specific host-individual. When a host is infected with a virus such as HIV or influenza, the virus rapidly makes many copies of itself. Because replication is imperfect, many of the copies of the virus will contain multiple point mutations [Sanjuán et al., 2010]. Some of these variants are advantageous and accumulate within the virus population (e.g., variants that evade the host's immune response). New deep sequencing technologies enable surveillance of viral genomes throughout entire populations. While workflows currently exist for identifying correlation between two genomic positions, the analysis process is a manual effort and prone to errors. Better analysis tools and support are needed to rapidly identify significant co-occurrences of mutations in genomes.

Our contribution is a design study (see Sedlmair et al. [2012a]) of the rapid identification of correlations between mutations in populations of a viral genome, where technology has become available to understand the population dynamics of viruses. We provide a

characterization and abstraction of the problem, allowing us to propose a solution for the generalized problem. We consider standard encodings for sequence and correlation data, and explore their use in an initial prototype. Through a participatory design, we reconcile failures in early prototypes and iterate on our design to better match virologists' needs. We assess this system through two case studies, and end with a discussion of the lessons learned through the problem characterization and the design study.

6.2 Biological Background

Our work is a part of an established collaboration between virologists and computer scientists to develop better tools for understanding the genetic mechanisms involved in viral infections. Team members from both backgrounds have worked together to build an understanding of problems, and have evolved tools that address them. Here, we describe the general problem of understanding viral population dynamics and the need for new tools to examine co-occurrence in this domain.

For the purposes of this chapter, the key biological concept is that the genome replication process of RNA viruses (e.g., HIV, influenza) is highly error-prone, resulting in the incorporation of random mutations of nucleotides throughout the viral genome. In an infected host, HIV and influenza exist as a diverse collection of similar yet distinct viral particles, each with its own genome. While most mutations in RNA viruses are catastrophic to the continued survival of the virus, those mutations that are beneficial to viral fitness continue to propagate. Generally, the longer a virus has infected the host, the more variation in the viral population.

Identifying combinations of mutations (*co-occurrences*) in the viral genome is critical for understanding important biological functions. For example, simultaneous mutations at three or four positions on an external viral protein haemagglutinin (HA) of an avian H5N1 influenza virus permits transmission to mammals [Imai et al., 2012]. Interestingly, these mutations do not confer transmission individually, but rather they need to exist together on the same viral genome (a concept named *epistasis*). Epistatic mutations are co-occurring mutations that, together, can allow a new biological function. Identifying co-

occurring mutations from virus populations allows for detailed characterization of genetic diversity and accurate assessments of viral function. A global view of co-occurrence can help understanding of how a virus works at a high-level, and serves to target *in vivo* experimentation of viral activity of larger epistatic interaction.

Nucleotides that mutate can cause the functionality of a virus to change by affecting emitted proteins. Regions of the genome that code for proteins are called open reading frames (ORFs), where a reading frame is a particular sequence of codons, which themselves are triplets of nucleotides. The translation from codons to amino acids (the building blocks of proteins) is degenerate as there are 64 unique codons (4^3) and just 20 amino acids that can be represented by the genetic code. Therefore, a mutation in the genome does not necessarily confer a change in protein coding—these are instances of *synonymous mutations*. Identifying these synomymous mutations as not significant mutations are important to consider (though even synonymous mutations may have RNA structure—and thereby functional—implications).

The rapid identification of epistasis and characterizing the functionality of sub-populations remains a challenging task. New genomic sequencing technology allows for analysis of the diverse genomic populations and continued disease progression. In particular, Next-Generation Sequencing (NGS) analyzes millions of nucleic acid sequences simultaneously, enabling detailed characterization that captures the proportional presence of viral sub-populations in a sample. The output of the NGS system are aligned sequences of short-read data—see Figure 6.1 for an abstract representation. These reads (on the order of hundreds of thousands) have associated start points in the global genomic sequence space. Due to limitations in current technology, however, only 300–500 nucleotides can be called for each read, limiting the range of co-occurrences that can be observed in pairwise genomic space. Newer techniques, such as including analysis from "paired reads," can increase this bandwidth, but still represents a hard limit on analyzing distant pairs in the genome.

6.3 Problem Details and Requirements

The process of discovering these co-occurrences of mutations in viral populations is not well-supported by any existing tool. Current workflows for discovering sub-populations demand either expensive processes examining all potential combinations, manual curation and exploration through the data using tools such as Microsoft Excel, or line-by-line inspection of aligned reads in programs such as Geneious Pro [Kearse et al., 2012], CLC Genomics Workbench [CLC bio], or the Integrated Genome Viewer [Robinson et al., 2011].

Our discussions with virologists identified two main analysis goals. The first is an idea of **diversity**: a better understanding of the amount of variation in sequence space. For example, higher variation within a population could indicate there are environmental pressures (e.g., an effective immune response) that is forcing the virus to diversify to survive. The second insight regards **functionality**, where the population of viruses can be separated into sub-populations that share coordinated mutations. This separation can provide researchers with a vector of attack to characterize the viral sub-population *in vivo* to see if a functionality shift is occurring.

The general problem is to identify pairs of genomic positions where mutations are observed to co-occur together. If we think of the reads (rows in Fig. 6.1) as observers making measurements about events in a global context (columns in Fig. 6.1), we can begin to determine how these observers connect these events. To understand the correlation of events, we can gather statistics from pairs of positions that are observed together—we call this *observer continuity*. In contrast, looking at observations without regard of observer continuity reduces to an independent event comparison problem, which is supported by existing visual analysis techniques for time-series data (*cf.* Javed et al. [2010]) or existing metrics (such as mutual information [Steuer et al., 2002]).

From this problem characterization, and through iteration and discussion with our collaborators (see §6.6.1), we collected a series of analysis tasks. The first (**T1**) is to *identify* significant co-occurrences of mutations. Virologists must be able to *explore in detail* a co-occurrence pair (**T2**), evaluating whether the particular correlation is important and requires further research. Important co-occurrences within the entire genome must be

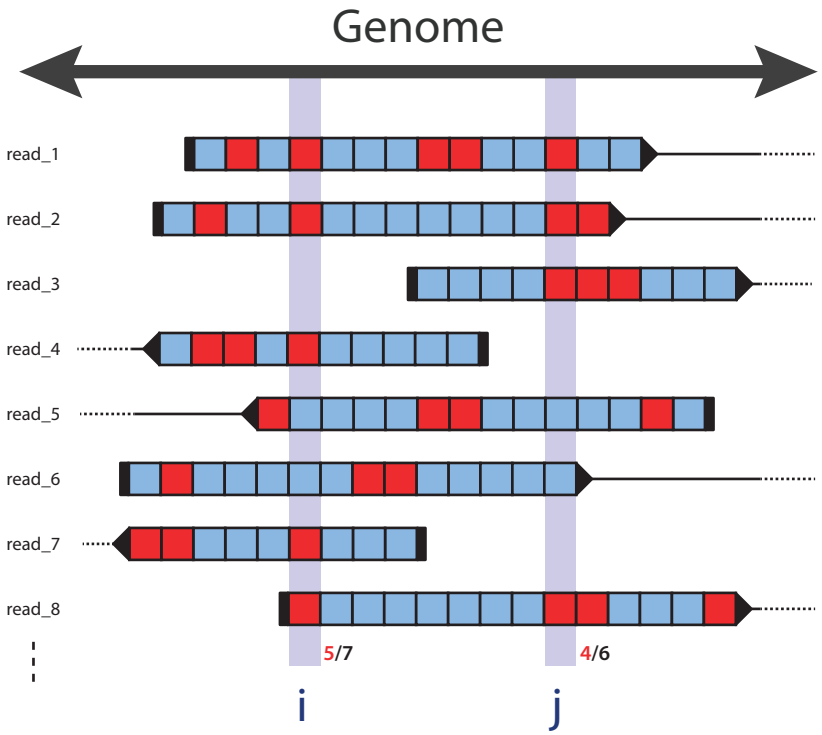


Figure 6.1: A visual abstraction of viral genomic data, where red boxes denote nucleotides that do not match the reference genome. Rows are individual reads from NGS, while columns are genomic positions. Two positions (i, j) are checked for mutation co-occurrence.

easily *summarized* (**T3**), requiring overview of all significant correlations in the genome.

We collected additional requirements based on the specific task domain. The presentation of the data in the visualization must align with the analysts' existing mental models of genomic data by (1) always presenting data in genomic sequence order (**R1**) and (2) displaying annotations alongside the genome to provide wayfinding for the analyst (**R2**). We found through discussions with virologists that a mental 'map' helped to orient themselves while navigating the viral genome. To be able to discover significant co-occurrences, there needs to be a scaffold to navigate the space of all pairwise correlations (**R3**). Finally, the visualization must scale to the typical dataset size (**R4**): hundreds of genome positions and hundreds of thousands of individual read segments, while remaining interactive to the anaylst in a web-browser-based deployment (which simplifies sharing of datasets).

Our approach to deal with the vast space of correlations is to define interest metrics to aid in filtering. Discussions with stakeholders suggested that there are a variety of

factors to consider in developing such metrics. The simplest of these measures is *positive correlation*, which can indicate potential epistatic mutations. The inverse, *negative correlation*, can also be interesting, demonstrating that combinations of mutations can be catastrophic to viral fitness. Secondly, there may be issues with *coverage*, where there may not be enough observations relating two positions to make significant judgments about correlation. Finally, the base rate of mutations at a particular position must be over the error rate of the NGS sequencer to be significant, otherwise spurious correlations that are misaligned may be counted as significant. We elaborate on these metrics in Section 6.5.

6.4 Related Work

6.4.1 Visualizing genomic data

There are many genomic data viewers that support the visualization and analysis of variants (see Nielsen et al. [2010] for a broad survey). The most common of these analysis tools are genome browsers, which juxtapose the raw genomic sequence alongside supplemental data, such as computational predictions and homologies. There are many examples of these tools, each of which are specific either to a particular task (e.g., resolving reads from NGS data [Fiume et al., 2010]) or a particular biological domain (e.g., cancer [Dees et al., 2012] or humans [Kent et al., 2002]). Although there are many browsers, most make assumptions that break our model of multiple, competing viral genome populations. For example, the MuSiC system [Dees et al., 2012] contains functionality that identifies statistically-probable correlations of mutations [Ding et al., 2008]. In particular, the use of fixed statistical judgments and sub-sampling methods are not well-suited to analysis of a viral population, as it assumes that non-matching reads are errors instead of an indication of a sub-population.

Specialized genomic visualizations can make visual encoding decisions that directly support particular analysis tasks. These systems either expose trends and relationships between annotations [Van Brakel et al., 2013], between variants and annotations [Ferstay et al., 2013, Demiralp et al., 2013], or between alternative splicing of genes [Strobelt et al.,

2016]. Many of these systems directly encode correlation. COMBat [Van Brakel et al., 2013] uses a re-orderable matrix view to highlight correlation between annotations, intentionally scrambling the genomic axis. Variant View [Ferstay et al., 2013] uses tracks to show overlapping annotations, as well as concise glyphs to convey information on types of mutations at particular positions. DecisionFlow [Gotz and Stavropoulos, 2014] allows the analyst to drive exploration through a large electronic health record space and presents health outcomes in Sankey-like diagrams, while Vials [Strobelt et al., 2016] uses a common genomic axis to ground analysis of splice groups. While some of these tools violate several of our initial requirements (e.g., COMBat violates **R1** and **R2**, Variant View does not scale to the data scales needed in this application **R4**), they provide precedent for the visual support of our three tasks (**T1–3**).

Many solutions for analyzing viruses, like Alvira [Enault et al., 2007], use a 'scaffold view' where sequencing reads are stacked atop one another, mutations are highlighted, and frequency of variants is highlighted by proportional sequence logos. These visual encodings have notorious disadvantages, including inablility to scale and and potentially skewing proportionality judgments (see Maguire et al. [2014] for a discussion), suggesting a more principled ensemble encoding. Similar to our system, LayerCake [Correll et al., 2015] supports finding variants between multiple aligned samples of populations of viral genomes by using color as an ensemble encoding, compressing horizontal space by binning positions together but otherwise maintaining strict sequence order. LayerCake highlights population dynamics only between viral samples, not within a particular sample. Therefore, LayerCake does not support discovering correlations between mutations as there is no notion of observer continuity.

6.4.2 Visualizing correlation

Visualizing correlation between events is a task of substantial interest in the visual analytics literature. Two primary methods of visualizing relationships between elements are through node-link and matrix-based visualizations (see Ghoniem et al. [2004] for a discussion). While node-link visualizations have issues of scale with increasing number of nodes, they are invaluable for analyzing multi-stage connections. On the other hand, matrices excel at

larger number of connections, though they suffer at providing aggregate judgments (*cf.* Díaz et al. [2002]).

Several studies have modified the typical uses of node-link and matrix-based visualizations to uncover trends in combinatorial relationships. Henry and Fekete [2006] use a matrix view in conjunction with a node-link view to better support analysis tasks of social network connections between individuals. Dunne and Shneiderman [2013] use aggregate glyphs to represent common visual patterns in node-link diagrams, managing complexity in the number of elements and connections shown. Other visual methods such as parallel sets [Bendix et al., 2005], parallel coordinates [Inselberg, 1997], and Sankey diagrams show how similar elements relate to one another through many continuous or categorical dimensions. These methods are helpful for conveying a general sense of how a subset interacts with different data dimensions, and we use parallel sets to visually communicate the level of correlation in a co-occurrence pair.

We use general trends found in these works to inform our own design decisions. For example, we anticipated in early designs that a matrix view would be a good way to reorder positions to identify significant co-occurrences. This led us to the requirement of maintaining genome continuity (**R2**) in order to support the virologists' mental models, upon which we elaborate in Section 6.6.

6.5 Interest Metrics

To reduce the correlation space that an analyst needs to explore, we identified the following three metrics that capture the intuitions of our audience for what is considered an "interesting" correlation. The first is *coverage*: we must have a sufficient number of the events in order to be confident that the measures we receive are not due to sampling error or noise. The second is *variation*, where each of the two sites must have a sufficient diversity of observations. The third is a *metric of co-occurrence*, which quantifies how unlikely is the relationship between the two sites relative to chance, given what would be expected by the statistics at each individual position under an assumption of independence.

Abstractly, we consider the set of events **E** in a data sequence, and observers **O** that

make observations about those events. Each observer O_k therefore represents a set of observations of the form $\{(i, +), \cdots (j, -)\}$, where each tuple contains a reference to an event in **E** (e.g., position i) and a categorical observation (e.g., $+$)—for example, if a mutation is observed at this position or not (a tuple is a square in Figure 6.1). Throughout our notation, we use Q as a collector of observers that have made a given observation about an event.

These metrics are summarizations of a co-occurrence pair, but do not individually confer a clear indication of significance. In different situations, an analyst may have different considerations. Therefore, we allow the user to dynamically set thresholds for these metrics.

Coverage metric: The *coverage* metric C_i for a particular position i counts the number of observations made about a position and can be used to determine coverage in comparison to other positions. C_i is computed by gathering all observers $B \in O$ that reference the position i and counting the number of observations in the returned set.

$$C_i = |Q_{(i_*)}| = |\{B \in \mathbf{O} \mid (i, *) \in B\}|. \tag{6.1}$$

We can extend this definition of Q to select sequences that have a particular type of observation at a position. As an example, $Q_{(i_-)}$ would match sequences that have observations at i that are negative.

Variation metric: The *variation* metric V_i can be used to threshold the prior probability for a variant to occur at a position. As an example, V_{i_-} below is the percentage of reads that are mutations at position i in our genomics context:

$$V_{i_-} = \Pr(i_-) = \frac{|Q_{(i_-)}|}{|Q_{(i_*)}|}. \tag{6.2}$$

Co-occurrence metric: Correlations that are interesting tend to be those where observations regarding one position seem to be conditionally dependent on the observation at another. To quantify this, we first count the observers of both occurrences. We augment Q again, capturing observations about a pair of positions, taking into account observer continuity—that is, an observation is only considered if and only if it contains data about both i and

j:

$$Q_{(i_-,j_+)} = \{B \in \mathbf{O} \mid (i,-) \in B \wedge (j,+) \in B\}.$$

Now, we can define a conditional probability. Let us assume that we are interested in the conditional probability that an observation is negative at position j given the negative observation at i:

$$\Pr(j_-|i_-) = \frac{|Q_{(i_-,j_-)}|}{|Q_{(i_-,j_*)}|}.$$

With these formulations, we can define a *co-occurrence metric* M_{i,j_*}.

$$M_{i,j_-} = \Pr(j_-|i_-) - \Pr(j_-|i_+) = \frac{|Q_{(i_-,j_-)}|}{|Q_{(i_-,j_*)}|} - \frac{|Q_{(i_+,j_-)}|}{|Q_{(i_+,j_*)}|}. \tag{6.3}$$

This metric is similar to metrics such as mutual information (see Steuer et al. [2002]). A key difference is that it takes account of observer continuity, allowing us to use conditional probability in our metric, in contrast to depending on joint probability (a potentially weaker assertion). Our metric also yields values in a fixed domain $[-1, 1]$, where -1 identifies strong negative correlation, 1 denotes strong positive correlation, and 0 implies no correlation. This is in contrast to mutual information, which has an unbounded, unsigned domain.

6.5.1 Interestingness in the virology problem

With Next-Generation Sequencing technology, researchers have the ability to understand the population dynamics of highly varying samples of viruses without the limitations of previous sequencing technology that would implicitly boost only the sequences with the highest occurrence. In engineering our solution, we decided to implement a pre-computation process that would compile counts of paired bases (all paired combinations of Q). With these precomputed counts, a front-end visualization permits interactive tuning of interest metrics. To determine mutations at nucleotide positions, we compare the collected data against a reference genome sequence. As our overall goal is to find co-occurrences of mutations, we de-emphasize the common case of reference-to-reference correlation, as this indicates the lack of any sort of epistatic functonality.

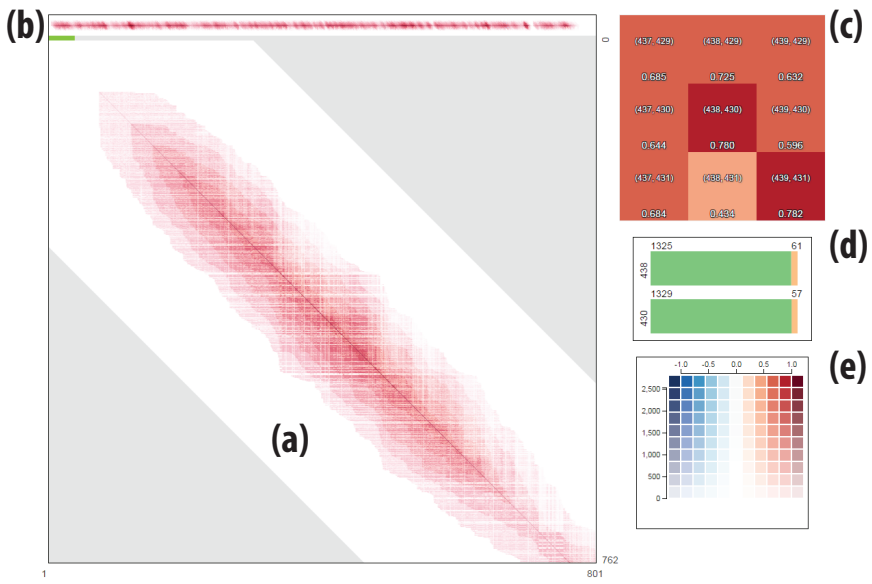


Figure 6.2: Our initial prototype to identify pairwise correlations between all positions i (x-axis) and j (y-axis). The matrix view (a) shows these co-occurences, and the overview (b) provides a horizontal overview of the space. The super-zoom window (c) highlights the coordinates and co-occurrence metric currently under the cursor, while the bar chart (d) presents the proportion of reads at a selected pair of positions. The legend (e) presents the 2D color key.

6.6 Visualization Design

Here we will describe our experience designing a visual analytics solution for the given problem, first presenting our early prototype (§6.6.1) using a matrix-based solution. The failure of this initial prototype prompted us to derive task **T3** (supporting overview), and requirements **R2** and **R3** (wayfinding and tunable filter parameters). We describe the rationale for the designs, and some of our lessons learned (§6.6.2) in incorporating implicit assumptions of the analyst into requirements for the final design (§6.6.3).

6.6.1 Initial prototype: Matrix-based visualization

For our initial prototype, we developed a matrix-based technique for looking at the correlations of mutations between pairs of positions (§6.4.2, see Figure 6.2). The design was inspired by previous work that use matrices to communicate relations, which excel at displaying large numbers of relationships in comparison to node-link diagrams.

Each cell in the matrix communicate the level of mutation co-occurrence ($M_{i,j\_}$) at a pair of positions i and j. We use a bi-variate color ramp [Trumbo, 1981] to communicate the co-occurrence metric (a ColorBrewer red-to-blue diverging ramp [Brewer et al., 2003]) and the coverage (lightness attenuation in Lab color space), together identifying significant co-occurrence. Details are available through a linked "super-zoom" panel, which displays the metrics for a 3×3 area under the current mouse position. A bar chart (below) compares the mutations and reference reads at the two positions, and allows for conditioning on the nucleotide type.

We took advantage of the technological limitations of NGS, where direct correlations are limited to a window in the low hundreds of positions (the maximum read length). This produces a banded matrix about the diagonal, so we thereby limit navigation of the space to a one-dimensional diagonal pan and zoom to prevent getting lost in the data space. To overcome the technical limitation of loading millions of data points to the client and displaying them in a web-based interface, we used WebGL to load the data into buffers in the GPU and to render the matrix interface. Supplemental views such as the super-zoom were implemented using the D3 library [Bostock et al., 2011]. Using the GPU for rendering allowed for real-time navigation of a 20,000 $\times$ 20,000 cell-space, as well as interactive updates by modifying uniform variables sent to shaders (see McDonnel and Elmqvist [2009] for a discussion).

6.6.2 Lessons learned from the matrix-based prototype

This early implementation had several pragmatic problems for exploring NGS data. The visualization was overwhelmed by many false positive results at nearly every pair of positions—many co-occurrence pairs had a saturated color (see Figure 6.2) but were not significant in practice. Through iteration on this design with stakeholders and a root-cause analysis, we found that although the co-occurrence metric was high in magnitude, the overall proportions of variant reads at those positions were very small (on the order of 1–5%), even though they had very high correlation to other positions. Many of these reads were determined to be misaligned reads by the sequencer. In additional, simultaneously visualizing a third metric (variation) requires a tri-variate color map, which are considered

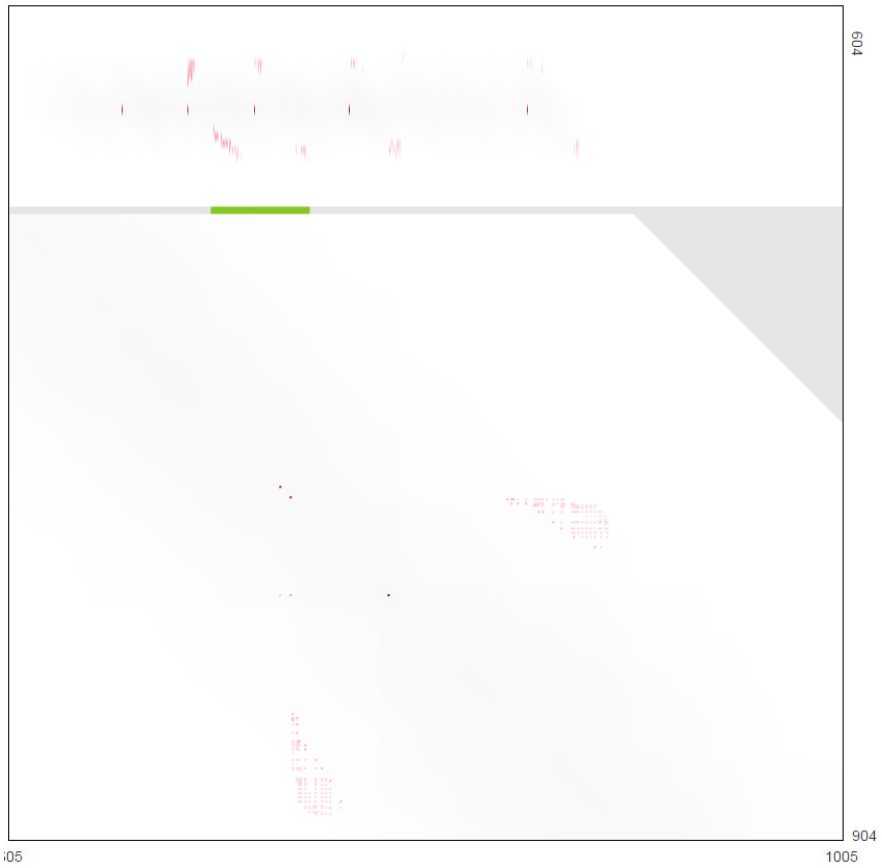


Figure 6.3: After filtering data points that fails the minimum variance threshold, the data in the early matrix prototype becomes sparse. Positions that have interesting co-occurrences cover very little visual space, making them difficult to find and highlighting the issues with the use of a matrix representation for our domain task.

to be impractical [Ware, 2009]. These constraints suggests an alternative method to filter out task-irrelevant co-occurrences (**R3**).

In order to assist analysts in identifying pairs of positions with significant co-occurrences, we added in a filtering gate to remove co-occurrence pairs where at least one position meets a minimum variant probability. This filtering made the data too sparse in the matrix to identify interesting co-occurrences (see Figure 6.3), suggesting task **T3**: providing overview.

While matrix re-ordering could help to emphasize correlation between positions, re-ordering the genomic sequence prevents analysts from leveraging their knowledge of particular sections of the genome, such as critical gene-coding regions. A requirement (**R2**) that emerged from discussion with collaborators was to provide a mechanism that exposed *annotations*, or interval identifers of the genome that provide a wayfinding mechanism.

They stressed that annotations can provide information on reading frames or identifying regions of interest. The overall difficulty of discovering interesting co-occurrences within the matrix view suggests a guided, interactive approach that does not embed relationships within the full data space.

6.6.3 CooccurViewer visualization

Our experience with the first prototype lead to a second design with revised tasks and requirements to support it (§6.3). Based on feedback from analysts and brainstorming potential solutions within our team and other virologists, we elected to modify our strategy to be driven by analyst focus (see Shneiderman and Plaisant [2015] for a discussion). To achieve this, we integrated our three tasks directly into the design (see Figure 6.4): a one-dimensional map that forms the overview and designates positions where interesting co-occurrence is happening (**T3**), metrics with which to filter the space of correlations (**T1**), and a detail view that describes the correlation between pairs of positions with explicit metrics (**T2**).

Overview

To support user-driven exploration of significant co-occurrence of mutations, we brought filtering to the forefront of the analysis. The virologist has the option to tune parameters of significance (§6.5), and only those correlations that meet the analyst-defined standards are displayed. The overview of these significant co-occurrences appears at the top of the visualization. Each position displayed has at least one significant co-occurrence with another position. These single positions are connected to their positions on the genomic sequence by gray wedges and are clustered together based on their proximity in genomic space. The overview can support up to 500 positions, but becomes more comfortable with less than 75 individual positions. Virologists using our tool to explore co-occurrences tended to tune the metrics until about 50 positions were visible in the overview.

The CoocurViewer overview includes a linear representation of the genome with annotation data. These annotations are represented by the colored bars above the genome axis

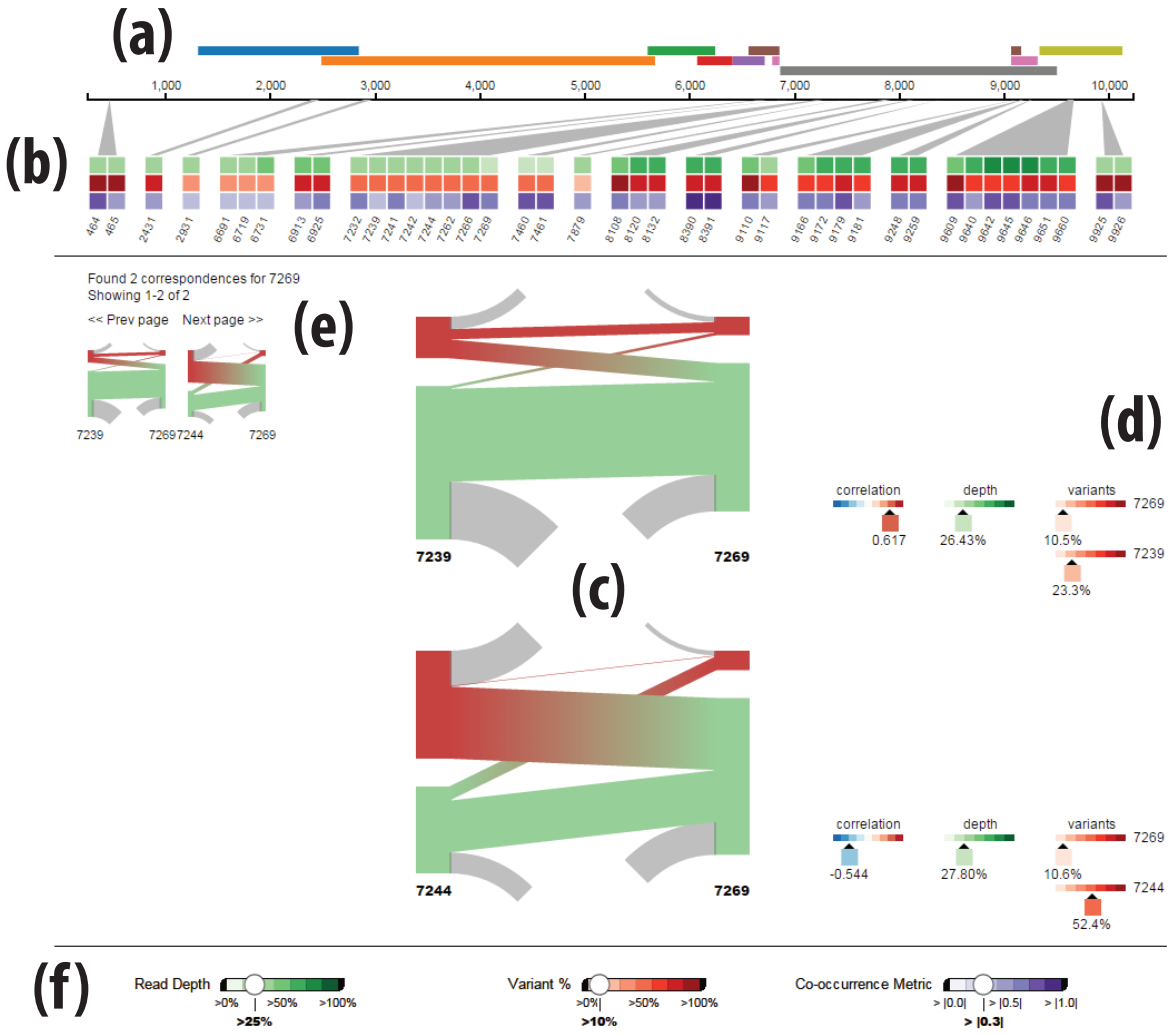


Figure 6.4: An overall view of SIV (§6.7.2) loaded into *CooccurViewer*. Annotations (a) denote regions of the genome that have some biological context, and the overview (b) denotes positions of significant co-occurrence, summarizing the three metrics (§6.5) using color. The correlation diagrams (c) provide a representation of correlation between pairs of positions, and some details (d) about metric values. The current position's summary of correlations (e) is given on the left, with small-multiple representations. The sliders (f) control the thresholds for the interest metrics and filters the co-occurrences shown in the visualization.

(Figure 6.4(a)), and provide virologists with biological context for positions in the genome. In particular, annotations marked as reading frames are used to determine if mutations within the region are synonymous mutations. Viewers are given the option of suppressing synonymous mutations, which treat those mutations as matching the reference genome. In practice, we found that virologists would activate this option to remove synonymous

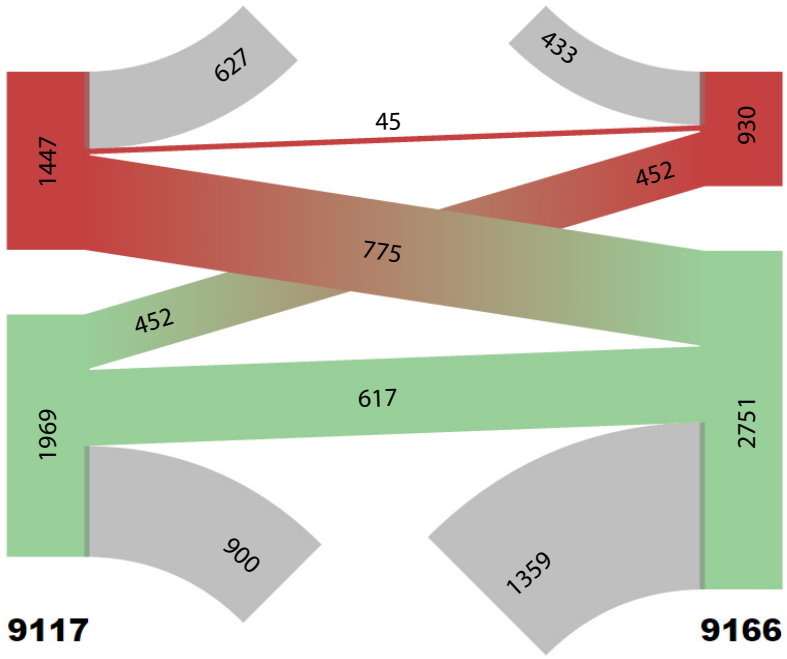


Figure 6.5: A close-up of a co-occurrence summary between two positions (counts included for explanation). The positions being compared are mapped to rectangles, with both reference (green) and variant (red) nucleotide types. The links show the correlated proportion of reads between the two positions. The gray arcs represent the proportion of reads that overlap one position but not the other.

mutations from display, but would also occassionally deactivate the option to identify mutations that could still have conformational implications.

Each position with significant co-occurrence is summarized by the three metrics introduced in Section 6.5, each color-encoded using separate ramps: coverage (i.e., *read depth*, in green), the base amount of variation at that position (in red), and the magnitude of the co-occurrence metric (in purple). In order to summarize correlations between multiple potential positions, the glyph at each position shows the maximum value of each metric independently. Sliders linked to these metrics (Figure 6.4(f)) allow the analyst to modify thresholds to filter out less interesting co-occurrences.

Co-occurrence Details

Once the virologist has selected a particular position of interest, the main view populates matching co-occurrences with that position. Through collaborative design, we developed a design to show "flow" between nucleotide types at two positions, similar to a version of parallel sets [Bendix et al., 2005] (see Figure 6.5). The connecting arcs show the proportion of reads (observations) that are one type at position i and are either the same or opposite type at position j. The gray arcs represent observations that exist at that position, but do not overlap the paired position. Tooltips can present more details on demand such as the number and proportion of nucleotide observations, including whether a particular nucleotide is potentially synonymous. For reasons of screen-space, only two pairs of co-occurrence detail can be shown at once, though all correlations for the current position are shown in a small-multiple display (see Figure 6.4(e)) and can be brought into full view by selection or through pagination.

6.6.4 Implementation

CooccurViewer is a system implemented in JavaScript, using the D3 library [Bostock et al., 2011] to map data to shapes on an SVG canvas. We use a pre-processing step to gather the 4×4 contingency tables (nucleotides at each position pair) from SAM files (short sequence read alignments) [Li et al., 2009] by comparing reads to a given reference sequence and counting paired combinations of bases for each pair of positions. We also compute the co-occurrence metric from these counts (see §6.5) and compile other data such as annotations. These data are packed into the binary files that are served to the visualization. This allows for minimal transport over the network, and the client-side nature of the visualization entails near-interactive rates for filtering the data shown to the viewer. CooccurViewer and the pre-processing library are open-sourced on GitHub and available at `http://graphics.cs.wisc.edu/Vis/CooccurViewer/`.

6.7 Case Studies

We present two case studies to demonstrate the utility of our visualization prototype. Through these examples, we illustrate how the visualization can expose significant correlation information. We show how the system is robust to displaying populations of viral genome samples in datasets of millions of pairwise correlations. We also highlight how our visualization design can help reveal new questions and insights about existing datasets.

6.7.1 Avian Influenza (H5N1)

In our first case study, different variants of the H5N1 influenza virus are explored. To understand the impact of within-host viral genetic diversity on replication and transmission of avian influenza viruses, Wilker et al. [2013] used deep sequencing to assess genetic variation from inoculated ferrets and ferrets infected via respiratory droplet transmission [Imai et al., 2012]. The authors reported that sub-populations present at low frequencies ($\sim$ 6%) could transmit via respiratory droplets. Interestingly, they showed that only one to two combinations of co-occurring mutations in the hemagglutinin (HA) gene were detectable early after infection in contact animals. Taken together, this shows that selective forces imposed a significant reduction in influenza genetic diversity during transmission.

We imported reference-based assemblies of the HA gene (1788 base pairs in length) from infected ferrets (six pairs, six samples each) into our pre-processing library. On average, each reference-based assembly contained 140k to 348k sequences (avg. 205k) and individual reads were 100 to 160 base pairs in length (avg. 149). Annotations denote regions in the sequence that code for the pre-processed HA protein (blue), a post-processed HA protein that becomes packaged in the viral envelope (orange), and a region on the HA protein that binds to host-cell receptors (green). A single sample's packaged data averages around 42MB.

There is a significant level of nucleotide variation near the receptor-binding domain of H5N1 viruses infecting ferrets. In Figure 6.6, using sequence data from a directly inoculated ferret sampled five days post-infection, there are a number of significant co-occurrences. Virologists focused on two particular positions with relatively higher amounts of nucleotide

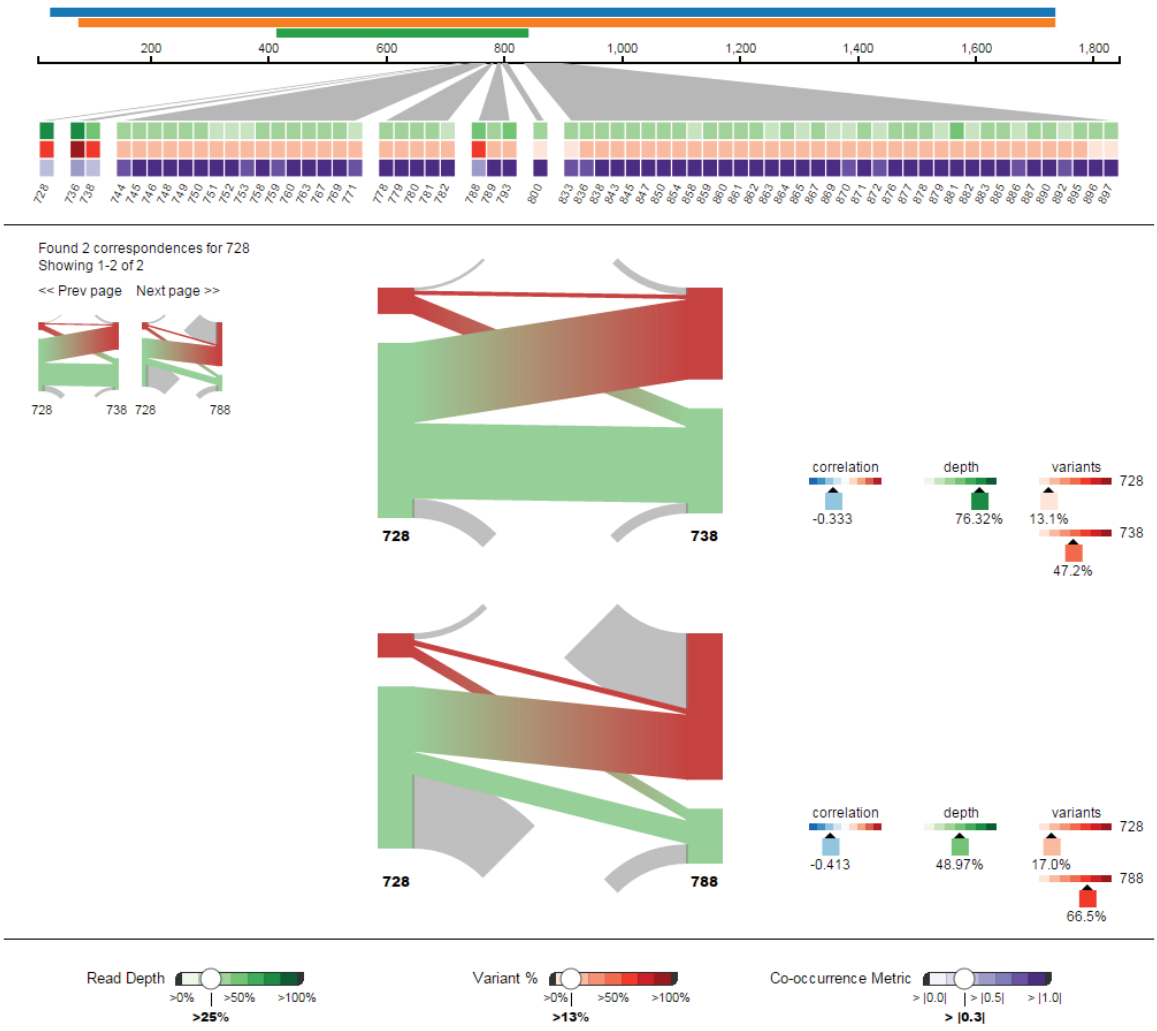


Figure 6.6: For this particular sample of an H5N1 viral population, a strong inverse correlation is identified between mutations at 738 to non-variant reads at 728, as well as a inverse correlation between positions 728 and 788, validating the results presented by the reference study [Wilker et al., 2013].

variability, where the summary glyphs are saturated red. Selecting position 728 (the farthest left summary), a strong inverted correlation is found between non-variant nucleotides at 728 and variants nucleotides at 788—this relationship was identified in the original study.

Through the use of the visualization, potentially interesting co-occurrences were readily identified. This is in contrast to the intensive, manual workflow used to identify co-occurrences in the original work [Wilker et al., 2013], which involved concatenation of all polymorphic sites and tabular exploration through these varying sites to find potential correlation (taking several weeks). The visualization, by contrast, specifically targets the

analytical task of rapidly identifying these interesting co-occurrences in the timescale of minutes.

6.7.2 Simian Immunodeficiency Virus (SIV)

SIV is a commonly-studied virus as an analog to HIV (human immunodeficiency virus). Variants accumulate during an HIV or SIV infection confer resistance to antiretroviral drug treatment or expand the range of cells the virus can productively infect. Understanding epistatic interactions are critical to target antiretroviral treatments. The dataset shown in Figure 6.7 comes from a macaque monkey 54 weeks after infection with a clonal, pathogenic strain of SIV [O'Connor et al., 2012]. In this case, we know the exact sequence and composition of the viral sequence (9,973 base pairs) that initiated the infection. The data contains 238k read segments, where each segment length is between 24 to 151 base pairs long (avg. 92). The 2.78 million pairwise count data and associated metadata is minimized to 170MB.

Virologists immediately saw from the summary (see Figure 6.7) that there is a high amount of variation in this particular SIV sample. Many of these significant correlations are inverse correlations, identified by a strong absolute co-occurrence metric (purple). In particular, virologists observed correlations in this dataset that may merit additional follow-up. First, there are comparatively few correlated variants in the structural proteins of Gag and Pol (the blue and orange regions stretching from positions 1309 to 5666). These are HIV/SIV genes thought to be under the greatest constraints; variation in these genes likely compromises the ability of the virus to replicate. The lack of correlated variants in these genes compared to the accessory and regulatory genes suggests that compensatory variants here are relatively infrequent. Second, they identified a cluster of correlated variants from nucleotides 9,609 to 9,660 that occurs within a known viral sub-population that is recognized by macaque CD8+ T cells. While it is known that the virus can evade detection by immune responses through mutations in this region, the virologists noted that examining the impact of correlated variants within this epitope may resolve sub-structure to the escape variant populations that would be missed with other analytic tools. The ability to foster these global insights demonstrates a remarkable improvement over virologists' previous manual workflows.

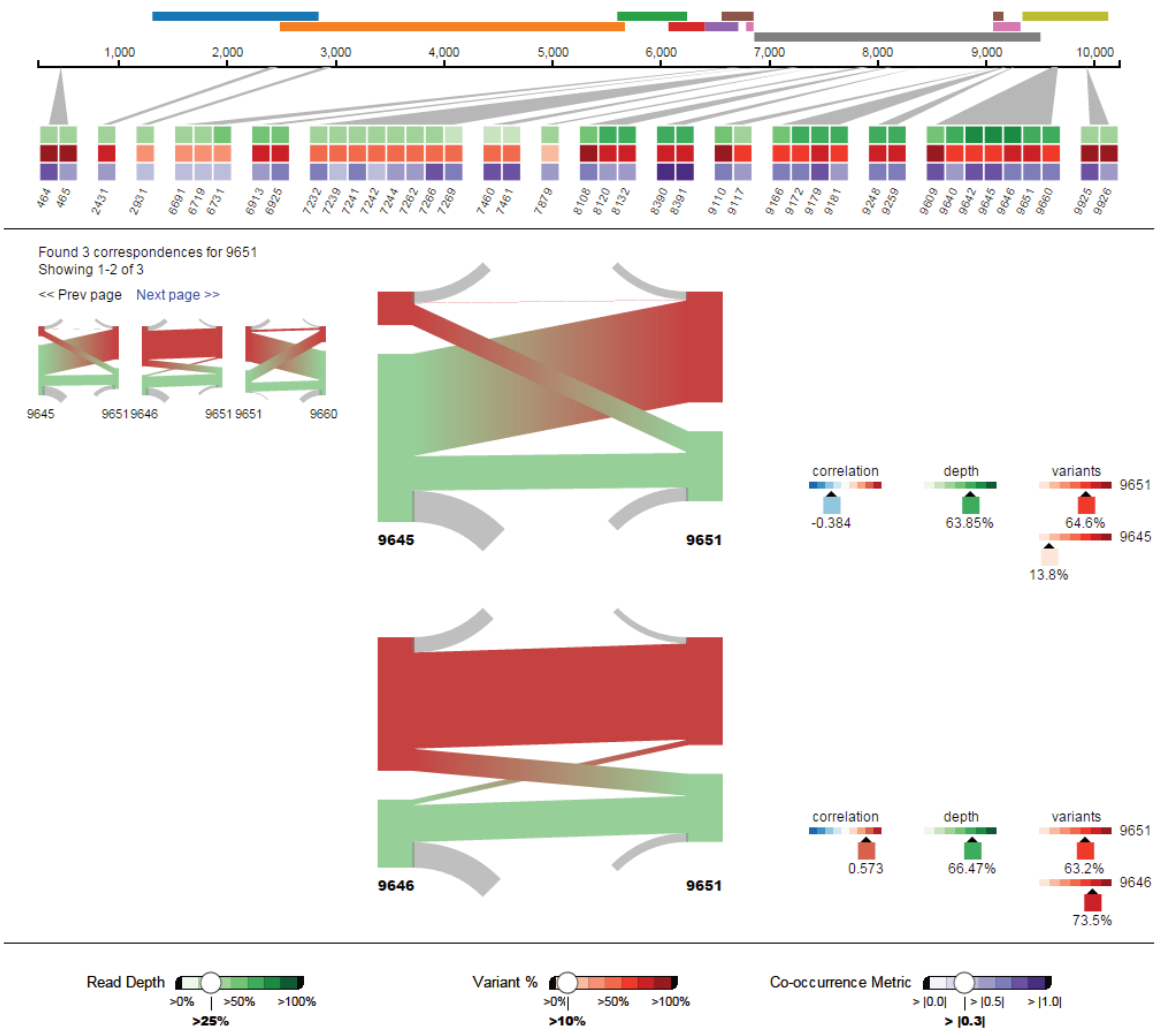


Figure 6.7: In this SIV sample, a cluster of correlated mutations appears within the Nef protein (top-right, dark yellow bar), known to harbor viral escape. Variants at positions 9,645 and 9,651 are inversely co-occurring with reference reads (mid-top), while reads at positions 9,646 and 9,651 are positively correlated (mid-bottom).

6.8 Discussion

Through this design study, we have learned several key lessons that generalize from our domain problem. Respecting the analysts' mental model of the anaylsis space and providing scaffolds for wayfinding proved to be critical in our design. We use a conjunction of multiple interest metrics to help narrow exploration in the large pairwise space of all pairwise correlations. We have also shown that combining multiple metrics through conjunction can help focus analysis when a single metric is insufficient.

In order to support analytical targeting for our design, we captured discrete components of significant correlations and generated definitions of these components. We quickly discovered that there was no single metric that captured if a co-occurrence between positions was significant or not, and elected to provide a mechanism to allow the analyst to select relevant thresholds dynamically. This interactive exploration affords analysis that can adapt to different analysts and datasets.

In this work, we focused on the problem of discovering co-occurrences of events within one sample of a population of viral genomes, and have shown it to scale to a significant amount of data (e.g., a viral genome 12k positions long with 250k reads leading to a ceiling of nearly 3 million potential co-occurrences). Extending our work to the problem of multi-sample comparison is important future work, though an independent problem. As an example, longitudinal studies of virus mutation usually span multiple timepoints, sometimes under different environmental or transmission conditions. While comparisons can be made implicitly between viral populations by switching the dataset shown in the visualization from one genome to another, it can be difficult to make explicit comparisons of correlation across samples.

The largest dataset we have supported thus far is the SIV dataset, which encodes 2.78 million 4×4 contingency matrices of pairwise correlations into our web-based visualization. We can scale to support the additional data of multi-level correlation (beyond pairwise correlation) and comparison across multiple timepoints by loading data directly to the GPU or offloading computation to a remote server [Moritz et al., 2015]. Applying data management principles such as indexing within the data (such as the *imMens* system [Liu et al., 2013]) could also increase data retrieval rates.

Finally, we have determined that our viral population dynamics problem is an instance of the abstract problem of understanding partially observed co-occurrences. This abstraction permits us to convey statistics and trends of co-occurrence events in a visual manner. The abstraction also allows us to generalize our work to other domains such as large-scale text analysis and time-series data, although our development of such applications is still in progress.

In this work, we have presented a design study for the rapid identification of correlated

mutations in populations of a viral genome. Through our characterization of the problem, we have identified requirements that led to metrics used to focus analysis on significant co-occurrences. We have shared our experiences in creating visualization prototypes to support our model task, demonstrated the effectiveness of our prototype design through our case studies, and summarized the lessons we have learned through this work. We hope to extend this work to higher-level correlations, and apply the lessons we have learned through this design study to other sequence-based data domains.

Beyond the application to sequence-based domains, the lessons in summarizing co-occurrence data have strong engineering and design implications. The decision to pre-process the data before visualization is a common decision, but the requirement of tunable parameters entailed on-the-fly computation from the core data source, necessitating transferring the full gamut of data. These data and viewer requirements helped to guide the iterative design process, resulting in the final design for *CooccurViewer*. The next chapter discusses engineering programmer abstractions for visualizations in a more general case, including support for scenarios with large amounts of data.

There are many challenges for effective, scalable visual representation of large datasets. Many of these core challenges for creating effective representations lie in creating scalable visual designs as well as efficient implementations that allow for interaction. Scalable visualizations allow the viewer to obtain an overview of the trends in the dataset, while interactive elements (e.g., zooming, expanding particular trends) allow the viewer to recover individual details upon closer inspection. Interactive methods and implementations are needed in order to tackle the challenge of aggregating and summarizing many elements while remaining interactive and comprehensible to the viewer.

With better guidelines regarding effective visualization, programming abstractions that promote effective design can help the visualization design process, enabling the use of exploratory visualization in their analysis. In this chapter, we describe lessons learned about the design of visualization programming interfaces, including creating the abstractions necessary to interface with underlying graphics interfaces and technology (WebGL) and the abstraction of concepts for the visualization designer. With this abstraction, practitioners are shielded from the nuances of the graphics interface, and are free to concentrate on high-level design decisions. Here, we start with a discussion of using WebGL as a tool to dynamically transform data relative to the current view, transition to discussing visualization-focused interfaces for WebGL, and conclude with a discussion of the `d3-twodim` scatterplot library, which focuses on providing scatterplot-like designs for the d3.js ecosystem [Bostock et al., 2011].

7.1 Using WebGL: Implementing the Splatterplot System

With WebGL, information visualizations can use the power of the client's GPU to bring interactive speeds to the scalable display of data contained within the ubiquitous nature of the internet browser. However, implementing visualizations in this environment imposes constraints, from the comparatively slow performance of JavaScript to the communication pipeline between JavaScript and the GPU, both of which require additional consideration. In

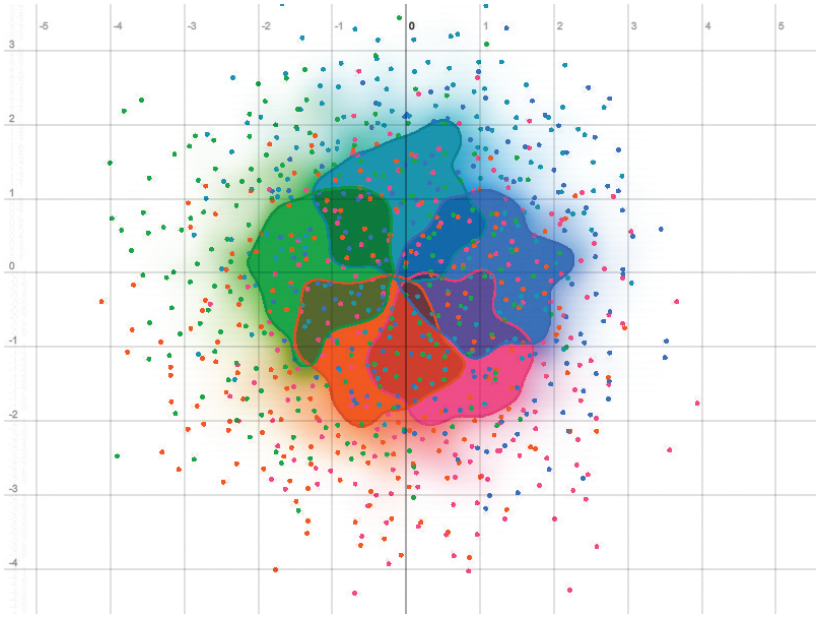


Figure 7.1: Our WebGL implementation of the Splatterplot technique [Mayorga and Gleicher, 2013], showing a subsample of five different sampled Gaussian distributions (about 7.5k points per series). The web-based implementation allows for interactive exploration of hundreds of thousands of two-dimensional points.

this section, we present our experiences in implementing the Splatterplot system [Mayorga and Gleicher, 2013] for WebGL (named *SplatterJs*), with discussion on how we worked within constraints for maintaining client interactivity in the browser (see Figure 7.1).

In our iterative development of SplatterJs, we ran into several pitfalls when porting the native-code OpenGL implementation to one using WebGL. One of the most significant issues was the amount of CPU-based computation done in the original model, which had a significant adverse affect on performance when directly ported to JavaScript and WebGL. This forced us to re-evaluate how we performed operations on the data, including the consideration of moving these computations to the GPU through the use of WebGL. In order to perform reduction and subsampling operations, we used general-purpose GPU (GPGPU) algorithms and store the results to textures to be used in downstream rendering steps. We expand on the specifics of the WebGL implementation of SplatterJs, and then expand on general lessons from our experience using WebGL to architect visual scalability for information visualization.

7.1.1 Architecting Visualizations for WebGL

The GPU (graphics processing unit) is a powerful piece of hardware that excels at the massively parallelizable operations such as determining the color of each pixel, given many inputs. The power of the graphics primitive pipeline to take data through programmer-defined vertex shaders, fragment shaders, and various compositing operations is a convenient tool to have in the visualization designers' toolbox. The allure of WebGL is in the marriage of GPU control coupled with the increasingly ubiquitous nature of internet browsers—it is an opportunity to bring GPU-accelerated graphics to the mainstream, without the overhead of installing a native application for the viewer.

WebGL itself is growing in popularity, thanks in part to the adoption of WebGL as the graphics standard for many mobile devices. Though the functionality in the adopted standard is considerably behind the current version of OpenGL (WebGL 1.0 currently implements similar functionality to OpenGL ES 2.0 [Khronos Group, 2015b]), the opportunity that WebGL presents by providing an interface to utilizing clients' GPUs as a computational unit is very promising for designing visualizations that can handle, process, and render constantly increasing amounts of data. Previous work has started to probe the utility of GL in general in information visualization, most notably the work by McDonnel and Elmqvist [2009] and Andrews and Wright [2014] that use OpenGL and WebGL shaders, respectively, to render common information visualization designs.

In typical programming practice, manipulation of the data (such as abstraction, filtering, projection, and subsampling; Chapter 3) is often done in CPU code, with familiar data structures. Once the data is transformed appropriately, it is handed off to the rendering procedures. In the browser, rendering can be done chiefly in one of two modes: those methods that bind data to shapes and modify those shapes depending on the data characteristics (e.g., creating an SVG drawing using D3.js [Bostock et al., 2011]), or methods that use raster-based methods (2D canvas or WebGL), which uses the GPU to compose the final visualization. In addition to the computations done on data to be visualized, methods also exist to operate over image space, which operate in the rendering step (such as color blending, depth-checking) over the shapes generated in an SVG image, or the

pixels generated on a canvas.

In natively-compiled code, data-space computations are typically done on the CPU, while image-space operations are often done on the GPU. This paradigm does not necessarily port well to WebGL—while the GPU is a powerful computational unit, the already-thin connection with which to repeatedly transfer transformed data becomes fragile within a web browser. To maintain responsiveness for the client, care must be taken to minimize the unnecessary transfer of data across this interface. Our experiences in porting the Splatterplot application to WebGL shows that minimizing this data transfer is key to maintaining interactive speeds for the user.

To minimize the transfer of transformed data from main memory to a GPU buffer, one possible solution could be to push computations to a backend server. While this operation may seem expensive, recent developments in HTML5 have enabled the transfer of binary data directly to WebGL through well-typed arrays in JavaScript called *arraybuffers* [Khronos Group, 2015b]. Arraybuffers can be filled manually, through XML HTTP requests (XHR), WebSockets (essentially TCP connections directly to the client), or WebWorkers ("multi-threading" for JavaScript) [Khronos Group, 2015a]. Using these interfaces (e.g., setting the `messageType` to `arraybuffer`), one could conceivably use a database or computational backend to stream new or transformed data directly to the client's GPU buffer for immediate visualization. Previous work has started to explore this space of loading data through well-formed blobs, such as the *imMens* system [Liu et al., 2013] that brings a data cube client-side through the loading of specially-designed PNGs to support interactive brushing and linking of large amounts of data, using the client's GPU as a processing unit. We note that this area is potentially ripe for additional work, and encourage the exploration of this space.

An alternative solution to moving data-space computation to the backend could be to move data-space computation directly to the GPU. The utility of this solution depends on the feasibility of porting the data transformation to the GPU, and managing the pipeline of data-space and image-space computations done on the data from data ingest to visualization rendering. The data-space transformations of abstraction, filtering, projection, and subsampling can be performed in WebGL using fragment shaders by employing GPGPU

algorithms and re-purposing image-space algorithms for data. For example, to find a maximum value in a dataset, we can use the GPGPU pattern of *reduction* [Buck and Purcell, 2004] to use `max()` to repeatedly reduce the values in a texture into a particular corner until one value remains (the maximum). We can subsample points by using the depth test, and compute distance fields using algorithms such as the jump-flooding algorithm Rong and Tan [2006]. We elaborate on this solution with a discussion of our experiences implementing Splatterplots in WebGL, noting the possible ways that data-space computation can be adapted to use the client's GPU.

7.1.2 Adapting Splatterplots to WebGL

Splatterplots [Mayorga and Gleicher, 2013] is an information visualization technique that deals with the issue of overdraw that occurs when plotting thousands to millions of individual points in a scatterplot. As an example, if many points occupy the same x- and y-position in a scatterplot, it can be impossible to distinguish whether one or many points are at a particular position, or even provide an idea of how many points are at that particular position (the concept of "clumpy" as discussed in §4.4).

Splatterplots deal with these issues of overdraw by utilizing kernel density estimation (KDE), which abstracts low-level features (individual points) to provide the viewer with an idea of the density of points in space. The key idea in Splatterplots is to use a screen-space KDE, which has the effect of performing abstraction at overview scales and revealing details at detail scales, while also highlighting representative outlier points outside of the thresholded region. These heuristics combine to create a visual paradigm that can handle visual scalability for scatterplots at a high-level overview, while also supporting interactivity (through its screen-space parameterization of its KDE) to recover individual points and positions at smaller scales.

The processing pipeline for every rendered frame in Splatterplots is shown in Figure 7.2. Each operation is colored based on which computational unit it utilized in the original implementation. For each data series, the points are drawn to a texture that collects the density of points on each pixel (*overdraw*). This density is then approximated by a kernel density estimation (KDE), using a Guassian distribution as the kernel. The maximum

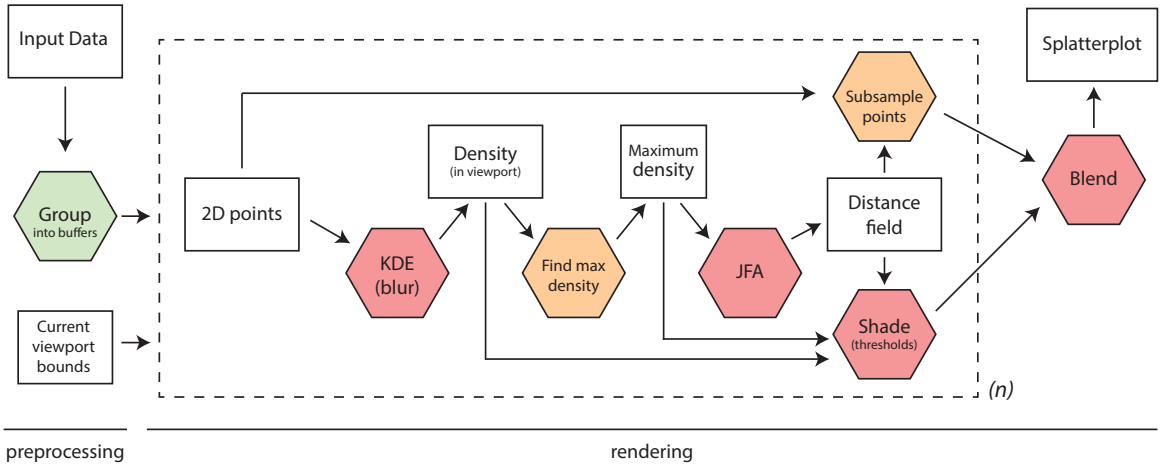


Figure 7.2: Data flow through the Splatterplot visualization technique. All processes (hexagons) produce outputs (rectangles), which take the form of textures. Those computations that are performed on the CPU in the original implementation are in orange, while GPU implementations are in red and the one-time preprocessing computation is in green. We adapted all rendering steps to the GPU, including rendering steps that transform input data for downstream rendering (§7.1.2).

density is recorded, a thresholded region is defined (by default, those pixels containing 50% of the maximum density), and representative points are randomly subsampled at regular intervals outside of the thresholded area to reinforce that data exists outside of the thresholded region. Finally, each series is composited together to form the final Splatterplot, using generated colors for each data series that are selected to be the most discriminable, and mixing in a perceptually-motivated way. Of particular note in our discussion here are the operations that find the maximum density value (an operation also done by previous visualization systems using density textures, *cf.* Gansner et al. [2011]) and the subsampling of points for drawing representative outliers.

Through these operations, many of them were suited directly for implementation in WebGL. For example, OpenGL is well-suited for collecting the per-pixel density for every pixel in the viewport: disable the depth test, enable blending, and change both the blend equation to add (`gl.FUNC_ADD`) and the blend function to one (`gl.ONE`), all of which results in full addition of pixel values in the fragment shader. Some operations, however, were conceptually easier to implement in native code, such as determining the maximum density (read the texture into memory and iterate through the buffer) or subsampling the points

(use a spatial data structure to iterate and subsample points to use as outliers). While potentially inefficient for performance, native code (the Splatterplot project was originally implemented in both C++ and C#) swallows the cost, and the technique remains interactive for the viewer when panning and zooming the dataset (see Mayorga and Gleicher [2013] for detail in the scale of the data).

After the paper was presented, the technique stirred interest, but potential users wanted an online solution that would let them quickly visualize their own data and see higher-level patterns. A natural choice to implement this system was WebGL, given the similarity between the GL interfaces. In the implementation of this system, however, several issues were encountered, and needed alternative implementations to maintain the real-time interactivity of the visualization. Most notably, performing transformations of the data (hundreds of thousands to millions of points) to support the various heuristics of Splatterplots in JavaScript proved to bog down clients on even the most advanced systems. Reading a (float-encoded) texture back into local memory is an illegal operation in WebGL 1.0 [Khronos Group, 2015b, §5.14.12], the method used by the original implementation (though WebGL workarounds exist). Randomly subsampling data points to select exemplar outliers iteratively in JavaScript proved to be too slow.

These sort of issues motivated us to explore re-architecting Splatterplots for WebGL. We appreciated not only having a completed prototype through this exploration, but also reusable components (such as the KDE implementation) for future visualizations using WebGL. From our experiences herein, we abstracted these rendering methods into two methods (`drawPoints` and `drawQuad`), discussed in further detail in Section 7.2.

Performing Data Transformations in WebGL

Here, we will describe some of these issues in detail and describe some of the WebGL techniques we employed to address these performance issues.

In determining how to threshold data series in Splatterplots, it is necessary to determine the maximum density value currently in view. Previously, density data for all pixels were read back into main memory, using an iterative search to find the maximum value, which was then used as a uniform parameter to downstream calls. Although WebGL does not

support the `readPixels` method to read values from textures with encoded floats, methods have been derived for encoding float values into four uint8 values of a RGBA texture according to the IEEE 754 specification (*cf.* Scheidegger [2015]), then calling `readPixels` on the surrogate texture to retrieve the original float value. We elected instead to use the common GPGPU design pattern of reduction [Buck and Purcell, 2004] that reduces values in a texture by consolidating values to a particular corner in order to find the maximum density value.

Given a texture and a step size, an aggregation measure (in this case, `max`) can be done in several passes over the texture. With a step size of 8 pixels, an 8×8 square can be minimized to a single pixel by repeatedly applying the aggregation function. For a canvas of 800×600 pixels, three passes (with a step size of 8) are necessary to reduce the nearly 500k pixels to two. Instead of passing a float uniform to downstream shaders that determine the thresholded region, the final max texture can be passed along, with subsequent shaders instructed to pull the value of the maximum density from the top-left corner of the reduced texture.

Representative outlier points are shown in Splatterplots to alert the analyst that data exists outside of the thresholded and shaded regions, even when viewing the dataset at an overview-level where the points would normally be blurred away. To minimize excess data display, only a single point is shown in every 25×25 pixel block (parameter-tunable). In the original native code implementation, points were iteratively picked at random at binned intervals in main memory. This approach did not scale when porting to JavaScript due to the high computational cost.

As all data points are retained in a data buffer in the GPU, we utilized a two-pass algorithm to (1) write point coordinates to a binned location in a temporary texture and (2) draw the point at the coordinates provided by each binned location. To select just one particular point from every grid cell, we associate each data point with a random value between zero and one and assign it to the z-coordinate with the depth test turned on. This has the effect of always selecting a single point for each spatial bin, as well as preventing twinkling (points winking into and out of existence) of individual outlier points when panning and zooming the display.

Implementation of WebGL Splatterplots

In the user interface of the WebGL Splatterplots application, we have added several sliders that allow the viewer control over the bandwidth of the KDE function, the threshold of the thresholded regions, as well as a outlier clutter metric (nominally the subsampling grid size). The event handlers for these elements modify the uniform parameters passed to the shaders and trigger a redraw of the canvas to interactively provide the user feedback when the slider is moved.

The application allows the viewer to upload their own data files, and asks for feedback when parsing a flat file for the two dimensions to plot (x and y dimensions), as well as an optional 'group by' column, which is used to separate the singular file into multiple data series. A working demo (allowing data uploads) and the source code of the WebGL splatterplots application are available online at `http://github.com/uwgraphics/splatterjs`.

7.1.3 Discussion

Through this section, we have discussed the use of WebGL for enabling web-based, interactive data visualizations that previously were only possible in natively-coded applications. Through several techniques of moving some data transformations to the GPU, we can empower viewers to use a complex visualization system without the additional cost of having to run and install software. A web-client's GPU can be utilized to transform data through aggregation and subsampling for use in downstream visual rendering.

The Splatterplot paradigm can even be used as part of a web mapping application (see Figure 7.3). The dynamic nature of the technique and the use of WebGL enables the visualization to remain responsive, even when the number of points number in the tens of thousands. As the threshold for dense, filled-in regions is dependent on the current maximum density within the current viewport, panning around a map can lead to discontinuity in the thresholded regions—they may sporadically grow and shrink. This can be resolved by computing and storing a maximum density value for every zoom-level, regardless of the viewport's position. Notwithstanding, the Splatterplot technique provides a view-independent interface to spatial data, and protects representative outliers

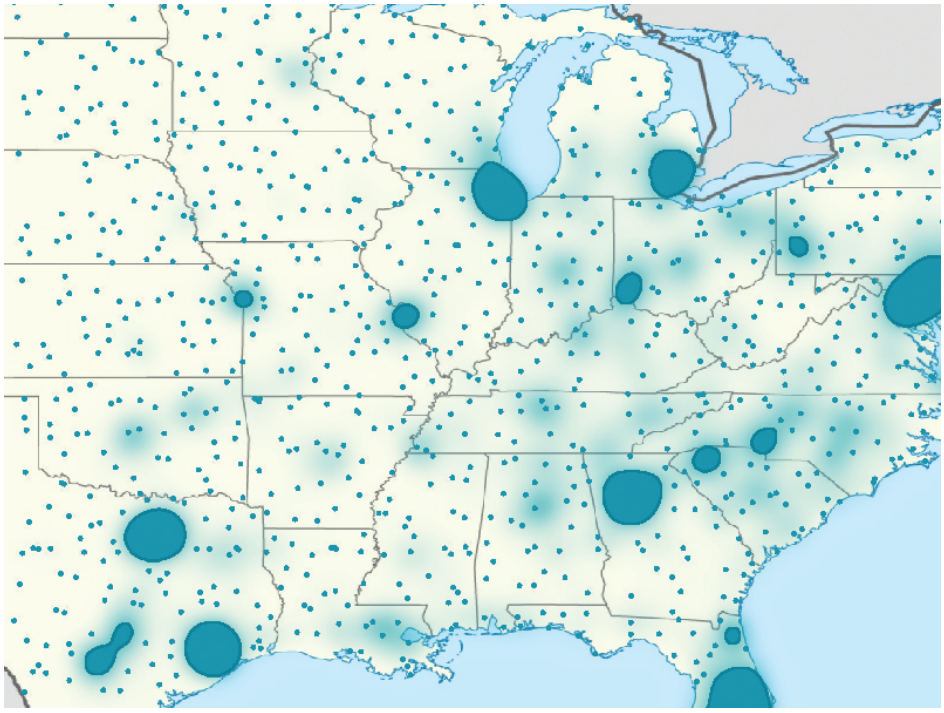


Figure 7.3: A single year (2011) of FARS (Fatality Analysis Reporting System) data shown in SplatterJs. The dataset comprises nearly 31,000 points. The viewer is free to zoom and pan about the dataset, much like in typical web-mapping applications.

by maintaining their display when zooming in for more detail.

WebGL has proven to be a very portable way to present and disseminate a data visualization. From our experience, however, there are several factors to consider when evaluating the use of WebGL in an information visualization. Chief among these factors is the reality that most of the data-space computation will need to be done in WebGL. While it may be more natural for the programmer to implement data-space operations using JavaScript, the nature of loading data repetitively from the client's browser to the client's GPU and vice versa has shown to be an expensive operation. If possible, all time-consuming data-space operations will have been done before WebGL receives the data, or these operations must be possible with vertex shaders. The method of delivering data to the client must be reliably quick—although we use flat files in this case, some of our other forthcoming work take advantage of binary interfaces for efficiently loading large amounts of data. Finally, the image-space operations required for visualization rendering must fit the GL paradigm; data must either be discrete and be aggregated for display by (multiple) shaders, or be

encoded in such a way that the data element maps directly to a graphics primitive.

We transform the data in our WebGL splatterplot implementation in JavaScript from uploaded comma-delimited files, but we also have had success in other applications using XHR requests for binary data (using `xhr.responseType = arraybuffer`) to fill well-typed arrays in JavaScript, and subsequently loading WebGL buffers with that data, though care must be taken to handle varying endianness of the data [Mozilla Developer Network, 2015]. As noted before, loading binary data to a JavaScript application is not just limited to XHR requests; WebWorkers and WebSockets can also handle binary data, which potentially enable receiving streaming binary-packed data from database sources. Additionally, using the `DataView` construct available in ECMAScript v5 allows for parsing of heterogeneous binary streams. We see this functionality in conjunction with WebGL's ability to handle streaming data as a ripe area for future exploration.

Although we have shown just two particular data-space implementations on the GPU, we believe that exploring the space of data transformation implementations in WebGL can help enable visualizations of larger scale and greater complexity in an implementation space more accessible to viewers. Through the use of WebGL, we can take advantage of parallel computation to compute per-pixel densities, and compose image-space representations to realize an interactive, scalable visualization. In particular for scatter data, the use of WebGL helps to take advantage of the collection of point densities, which can then be composited through multiple image-space operations to generate a Splatterplot.

To help generalize this methods for other visualizations, we have created a layer of abstraction around the main rendering operations used in SplatterJs, discussed in the following section. These methods include collecting point densities (`drawPoints`) and operating on the output(s) of previous rendering steps (`drawQuad`). The use of these methods only require the input of data and a shader program from the programmer—the necessary flags for enabling collection of frequencies and drawing full-frame triangles to enable proper texel coordinates are set within the methods. These two methods perform the necessary operations to set rendering flags (such as blending modes) and configure framebuffer and texture bindings without direct involvement from the programmer for these low-level details. These abstractions also help to abstract away historical WebGL implementation

details that may change over time.

7.2 `d3-twodim`: Scatterplot-like Designs in the Browser

We developed a d3.js plugin [Bostock et al., 2011] to abstract the programmer interface for exploratory scatterplots. The genesis for this project began with our collaboration with humanists, who were interested in using computational methods for "distant reading" of recently digitized, historical texts. These methods included topic modeling, which generate very high-dimensional vectors (30–150) per text [Alexander et al., 2014]. A common method for exploring these high-dimensional spaces involves using a dimensionality reduction (DR) technique such as principle-components analysis (PCA) or spectral analysis to display texts as points in a scatterplot [see Alexander et al., 2014, Fig. 7]. The scatterplot becomes a critical component to support exploration—specifically the development and confirmation of hypotheses, particularly supporting judgments of similarity between works that cluster in the visualization.

In recognition of the scatterplot's importance in DR scenarios (more detail in §4.2 and Sedlmair et al. [2013]), we developed a scatterplot-based library to support viewer exploration of data in two-dimensions. The development of this library focused on supporting viewer interaction, and therefore provides support for linked components such as legends, as well as direct interactions such as mark selection. In addition, the library provides programmer abstractions for constructing scatter-based visualizations using WebGL. With available hooks, external components such as data tables can connect the marks in the visualization to object metadata. The full implementation of the library is available as a plugin on GitHub: `https://github.com/uwgraphics/d3-twodim`.

Through the development and use of the library in internal settings, a common interaction paradigm began to emerge. A frequent method of interaction with the data would be to select a particular subset of the data, and this "highlighted" data compared against the base distribution of all points. To support this, `d3-twodim` supports message passing of `highlight` events between components—helping components respond to adapting the configuration to better support these subset views (see Figure 7.4). This highlight interaction

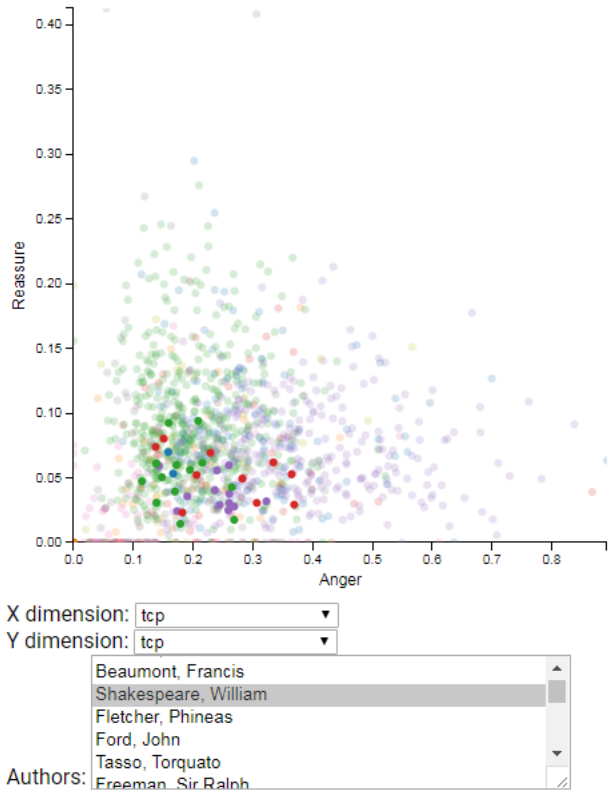


Figure 7.4: A subset of points highlighted over the full set of texts under consideration. Marks not in the highlighted set have a reduced opacity value, highlighting the set of marks under consideration. Click and hover events allow a viewer to recover metadata and data about individual marks through external components and pop-overs.

is baked into the library—all components must be responsive to this highlight mechanism and adapt to emphasize the given subset.

To remain flexible in the visual representation, multiple encoding types are supported. The modular design of the library permits programmers to add their own designs, given the data supplied through the library. As an example, a hexagonal binning implementation can replace the traditional circle-mark scatterplot. Multiple rendering mechanisms are supported, including 2D Canvas and WebGL. To support implementation of WebGL techniques, we generated a low-level interface for visualization designers to take advantage of WebGL. This interface provides only two drawing methods, but is sufficient to implement Splatterplots using the interface. The key insight here is to allow the programmer to manage components such as textures, buffers, and the current viewport bounds of the data, while abstracting the minutiae of blending methods and binding textures and buffers.

With this interface, the library allows the programmer to set data, define color ramps, and pass arbitrary shader code to the rendering methods. Utility code supports flattening data objects, allocating space for buffers and textures, and computing sensical bounds for the viewport. Once setting up the data source and other required buffers and textures, the programmer can choose to call a `drawPoints()` or `drawQuad()` method, passing in any uniform values, textures or buffers to bind, and the shader program to execute as an anonymous JavaScript object. The `drawPoints()` method draws data from the buffer based on two given dimensions for each data item, and positions the points based on the given viewport bounds (which is converted inside the WebGL shim to screen-space coordinates).

In order to support more complex rendering methods (such as for Splatterplots; see Figure 7.2), `d3-twodim` supports rendering directly to a texture. The `drawQuad()` texture supports reading textures from previous rendering passes as the "input" data, enabling pixel-by-pixel operations in the shader. This type of rendering is used extensively in the *SplatterJs* rendering pipeline, such as blurring density data, shading individual groups, and blending multiple series together. Though still operating at a low-level—the programmer is responsible for developing working GLSL shader code—these abstractions reduce the friction of dealing with functional WebGL code. As a proof-of-concept, Splatterplots have been implemented in `d3-twodim` using these WebGL abstractions.

7.3 Discussion

The two engineering projects described within this chapter have stressed the need and utility for methods that provide abstractions for visualization practitioners. These projects presents avenues of solutions that take advantage of rendering techniques for data processing (GPGPU algorithms) to make WebGL tenable for large data scale in the browser, and abstractions for presenting scatterplot-like designs. The `d3-twodim` library delivered herein helps to scaffold internal exploration and discussion of rich humanties data, providing a visualization front-end component to support computational exploration of literary data.

Usable tools with which to design and implement visualizations is often the goal of visualization researchers, but the engineering work tends to be undervalued in terms of

academic contribution. In response, much visualization work presents code or abstractions that work in a limited number of scenarios, generally limited to a particular data domain or analysis scenario. Here, we have attempted to provide abstractions that support a generalized set of scenarios and data, supporting the construction of scatterplot-like designs for exploratory analysis. With the contributions of earlier chapters in this dissertation, we hope to see such development of usable tools support a greater breadth of use cases, including analysis scenarios for the development of machine learning models and generally in data science. We expand on these ideas in the final chapter.

8 DISCUSSION

The use of overview to summarize data has been a long-standing goal of visualization [Card and Mackinlay, 1997]. Many different approaches to help scaffold the potential solutions have been proposed, either by minimizing the data before the visualization, or by novel visualization techniques that aggregate and transform the visual representation to show high-level information [Casner, 1991]. With the continued generation of data, scientists, professionals, and the general public need a consistent, accurate visual representation in order to interface with data that can affect their daily decision making.

In this thesis I have organized and categorized the methods and factors that lead to effective summarization of data. Using published papers in the data visualization research literature, we have gathered trends of different methods of data minimization and how it affects the high-level information communicated by a resulting summary visualization. Based on these results, we create an actionable framework for the effective use of scatterplot and scatterplot-like designs at scale, proposing a host of factors that affect design. These two organizations clarify the role of how factors can affect the type of summary afforded by a visualization, and provides avenues to continue to build upon these organizations for semi-automatic design selection for effective visualization.

We performed two case studies that utilized summary visualizations to deal with large amounts of biological data. In the two scenarios, we applied an iterative design methodology [Sedlmair et al., 2012a] to gather requirements, and derive designs that met those requirements. Iteration on these designs involved the collection of factors that prompt design decisions, including understanding how the scientists interacted with their data (what tasks did they perform?) and the characteristics of their data (what is complex about their data?). Within each case study, we describe how we made design decisions to address the concerns of scalability and complexity, and describe how these design solutions can be abstracted to similar problems in other data domains. We also describe how domain experts were able to use these visualization systems to derive new insights to their own data, helping to satisfy the goal of any visualization designer.

In the final chapter, we describe our early efforts to use these developed abstractions to

drive abstractions for programming and designing visualizations. We take advantage of the WebGL interface to use the graphics architecture as a parallelizable component that can perform data-space aggregation that is screen-space aware, and seek to abstract its usage through the `d3-twodim` library. With the two rendering methods (render buffer to viewport or rendering multiple "input" textures to the viewport), we can promote per-pixel computation to get the flexibility to define custom color space mixing and support high-performing multi-step rendering processes. These methods help to create web-based visualizations that scale to larger data volumes, while remaining responsive to interaction by the viewer.

8.1 Utility of the Proposed Framework

All together, this work informs how visualizations can be made to more effectively summarize large amounts of data for exploratory purposes. While the work presented herein presents a foundation for effective summarization of data using visualization, there remains a great deal of research to scaffold the design process for domain experts and visualization practitioners.

While the framework has not been realized into an available prototype for practical use, the organization presented within this dissertation both provides a process for visualization design and informs potential assistive tools for designers. The design process is a wildly open process, as noted by books on design [e.g., Williams, 2015] and for visualizations specifically [*cf.* Sedlmair et al., 2012a]. Such a framework informed by this work can quickly whittle away those combinations of design decisions that are not appropriate for the given analysis or presentation goals. The process of design rationalization can be supported by considering factors and attributes presented herein (purpose, data type, tasks to support, characteristics of the data to communicate).

Chapter 4 describes the relationships between the factors of task, data characteristics, and design decisions at a high-level. While not codified into a tool, the relationships between these factors described within the chapter can help visualization designers to select between methods. Figure 4.3 displays three types of scatterplot-like visualizations

side-by-side: a traditional scatterplot, a contour map, and a Splatterplot. The limtations of each design can be considered in order to support a more principled selection between these designs. As an example, the contour plot describes multiple levels of per-series point densities (additional fidelity to make more nuanced distributional judgments), but does not display outlier points that lie outside contained regions. By constrast, Splatterplots does display representative outliers (supporting tasks of identifying anomalies), but the design thresholds density to a single level, preventing viewers from discerning different levels of density within-series. While all design strategies have limitations, using the framework in this manner can help to make more principled decisions and rationalize choosing a particular visualization technique, given the analysis scenario at hand.

Chapter 3 describes these relationships at a high-level for summary visualizations in general. Chapter 4 describes these at a practical level, identifying combinations of factors that are or are not appropriate, given a potential design strategy. The scatterplot framework can be considered *actionable*—that is, several relationships between proposed factors have been proposed, and can be used to motivate the design rationalization process or populate the "business logic" of a potential designer-focused, visualization authoring application. The designer would retain ultimate control over the design of the visualization, but the factors such as task and data would help focus the attention on relevant design decisions.

The organizations proposed within Chapters 3 and 4 are derived through source material, and validated through statistical and logical means. The framework itself has not been applied and evaluated empirically with designers in practice. This validation would help to refine the grounded framework, and potential avenues toward this ultimate goal are described in the following section.

8.2 Future Work

The work in this thesis raises issues and highlights limitations of how we consider and evaluate the effectiveness of summary visualizations. While some of these issues are discussed within their respective chapters, provided here is a high-level outline of future work. The limitations of this work is discussed in the context of future work, championed

by research goals that are tenable in both academia and industry.

- **Task-Driven Design** — The first two chapters address the factors that address design, adapting the design process described in Munzner [2014]. These factors include the tasks that viewers perform with the visualization, and the characteristics of the data such as the dataset size, numbers and types of dimensions, and distributional qualities. While identifying these factors and the designs that support them, we do not provide an proportionate reductionist lens to the design decisions themselves. In the scatterplot design chapter (Chapter 4), we cluster design decisions based on their aggregate effects, but do not make judgments nor evaluate individual design decisions on their task support. We see this work as providing the holistic foundation for identifying the relevant factors for design, and future work would seek to quantify and clarify the support for different analysis tasks. This work can take the form of empirical evaluation with samples of the viewer population (the general public or other target audience) or by holistic evaluation of different versions of a visual analytics tool. With these results, we can then provide a matrix of design decisions and their support for tasks under different data characteristics, enabling speculative and semi-automatic design of visualizations.

  While we have developed such a list of factors for scatterplots, there are many other common visualization designs that do not have such tailored lists. Identifying common factors in data characteristics that lead to appropriate designs has been championed by Mackinlay et al. [2007] and used in the Tableau Desktop application. Rolling in viewer task to help identify appropriate design decisions could help generate more actionable visualizations for communicating and identifying anomalies and trends in a given dataset. Generating and disseminating this type of organization can help to organize (and discover) existing visualization techniques and identify areas of opportunity for novel techniques and combination of design decisions.

- **Validating Appropriate Design Decisions for Speculative Design** — Building off of task-driven design, semi-automatic and speculative scaffolding for the design process can help practitioners build more effective visualizations. By raising the floor

of effective design, such an approach can prevent domain scientists and practitioners from making critical mistakes that may misrepresent their data. Work such as Lyra [Satyanarayan and Heer, 2014] helps to support practitioners to create visualizations without programming, but do not scaffold the design process with an understanding of the factors that can affect design. By building factors into the programmer abstractions provided by a visualization library, or by building analysis scenarios into visualization design tools, more effective design defaults can be promoted. Work in this area would codify appropriateness measures from studies such as those presented in Chapter 3 and 4 into software interfaces, which would then reduce the space of design decisions available to the designer.

- **Validating Visualization Best-Practices** — "Best practices" with little empirical validation have long been embedded into the design principles for information visualization (see the discussion in Craft and Cairns [2005]). These practices are now possible to verify with improved evaluative techniques from the fields of perceptual psychology and human-computer interaction, such as crowd-sourcing perceptual experiments. While validating or discovering discrepancies in unconfirmed, anecdotal guidance may lead to new "tenets" of visualizations, work in this area also has the attractive advantage of lighting up under-explored areas of the design space, where no appropriate visual strategy exists for a particular combination of data, task, and data reduction technique. While the work presented within this document does not directly address longstanding best practices in visualization design, we supply the groundwork and organizations upon which to build new tenets upon.

- **Visualization in the Data Science Workflow** — The role of the human in the data science workflow has been somewhat understudied in the recent flurry of activity in machine learning. With current advances in information visualization, the work in this thesis addresses the challenges of using visualization for model verification, feature validation, and context-forming. With respect to the protein classifier work (Chapter 5), an attainable approach to these human-centered issues can be to treat the machine learning model as a black-box for extensibility. Learning method-specific

visualization has the risk of rapid obsolescence with the pace of technique innovation, though success stories exist (see §5.2.2).

Such a visualization will help one evaluate the output of the model in the context in which they are situated. How do the responses fare against human intuition? If discrepancies arise, can they be correlated with input features, the construction of the model, or correspondences within the data? Gauging the long-term vitality of long-running models (e.g., Facebook's News Feed or behavioral models) is also an important consideration for research, especially as visualization can play a critical role in discovering patterns through hypothesis formation, confirmation, and model exploration. Understanding human-centered processes in the iteration process and mapping these tasks to visualization-focused tasks can drive the development of tailored visualizations. The use of visualization would help to more intimately incorporate the viewer into the model development process.

- **Visualization for the People** — Data visualization has the unique ability to recast large amounts of complex data into a general, digestible format. While the work presented within this thesis is geared toward the "ideal" viewer or domain experts, the general public has much greater variation in experience (§2.3.1) and visualization literacy [Boy et al., 2014]. To ensure unbiased interpretation of graphics by a wide audience, more research is needed in adapting visualizations for individual differences. This evaluation would be in conjunction with evaluating the appropriateness of design techniques in a holistic manner with a diverse population. The ultimate goal in this space would be to *democratize* data visualization—to ensure that everyone can use visualization, and that everyone can design effective visualization.

  The phenomenon of data journalism, in particular, has begun to introduce visualizations into the public discourse. It is becoming more and more important, therefore, for the public to be able to evaluate and critique these visualizations, to both understand what the visualization shows and to build trust in the message that it conveys. Creating a scaffold for how we discuss and critique visualization will likely lead to the genesis of *critical visualization*, similar to the subfield of critical cartography,

where the message and context of a visualization are evaluated in situ. With the proliferation of open data, there is a great opportunity to enable greater participation and vetting of systems that are constructed to serve the public through visualization to equalize the power relationship: evaluating budget priorities, exposing services available to the public, and illustrating the potential ramifications of implementing laws. Visualization in this space would enable very unique opportunities for collaboration with local government, journalists, and social science researchers, as well as serving as a potential application for performing data science research for societal good.

The work presented in this document supports the thesis that effective design of summary visualizations can be determined by the methods of data reduction, viewer tasks, and data characteristics. By providing organizations and example case studies of exploration-focused visualizations, I have built a foundation upon which to build and concretize this theory of task- and other factor-driven design of summary visualization. The greatest challenge is to bring these organizations and guidance developed to support the design of visualizations towards addressing current challenges in science, communication, and decision making.

D. Albers, C. Dewey, and M. Gleicher. Sequence surveyor: Leveraging overview for scalable genomic alignment visualization. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2392 – 2401, Dec 2011. DOI: 10.1109/TVCG.2011.232 [*Cited on* pages 52 and 93].

D. Albers, M. Correll, and M. Gleicher. Task-driven evaluation of aggregation in time series visualization. In *Proc. Conference on Human Factors in Computing Systems*, pages 551–560. ACM Press, 2014. DOI: 10.1145/2556288.2557200 [*Cited on* pages 5, 22, 23, 93, and 96].

E. Alexander and M. Gleicher. Task-driven comparison of topic models. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):320–329, 2016. DOI: 10.1109/TVCG.2015.2467618 [*Cited on* page 86].

E. Alexander, J. Kohlmann, R. Valenza, M. Witmore, and M. Gleicher. Serendip: Topic model-driven visual exploration of text corpora. In *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 173–182. IEEE, Oct 2014. DOI: 10.1109/VAST.2014.7042493 [*Cited on* page 145].

R. Amar, J. Eagan, and J. Stasko. Low-level components of analytic activity in information visualization. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.*, pages 111–117. IEEE, 2005. DOI: 10.1109/INFVIS.2005.1532136 [*Cited on* pages 14, 30, 36, 62, and 76].

K. Andrews and B. Wright. FluidDiagrams: Web-based information visualisation using JavaScript and WebGL. In N. Elmqvist, M. Kennedy, and J. Hlawitschka, editors, *EuroVis—Short Papers*. The Eurographics Association, 2014. DOI: 10.2312/eurovisshort.20141155 [*Cited on* page 136].

N. Andrienko and G. Andrienko. *Exploratory Analysis of Spatial and Temporal Data: A Systematic Approach*. Springer, Heidelberg, 2006 [*Cited on* pages 14, 15, 24, and 30].

N. Andrienko, G. Andrienko, and P. Gatalsky. Exploratory spatio-temporal visualization: an analytical review. *Journal of Visual Languages & Computing*, 14(6):503–541, Dec 2003. DOI: 10.1016/S1045-926X(03)00046-6 [*Cited on* page 16].

F. J. Anscombe. Graphs in Statistical Analysis. *The American Statistician*, 27(1):17, Feb 1973. DOI: 10.2307/2682899 [*Cited on* page 21].

D. Ariely. Seeing sets: representation by statistical properties. *Psychological Science*, 12(2): 157–162, 2001. DOI: 10.1111/1467-9280.00327 [*Cited on* pages 21 and 93].

S. Bachthaler and D. Weiskopf. Continuous scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1428–1435, 2008. DOI: 10.1109/TVCG.2008.119 [*Cited on* page 60].

F. Bendix, R. Kosara, and H. Hauser. Parallel sets: Visual analysis of categorical data. In *Proc. IEEE Symposium on Information Visualization*, number 1, pages 133–140, 2005. DOI: 10.1109/INFVIS.2005.1532139 [*Cited on* pages 118 and 127].

H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–242, Jan 2000. DOI: 10.1093/nar/28.1.235 [*Cited on* pages 5 and 94].

J. Bertin. *Seminology of Graphics: Diagrams, Networks, Maps*. The University of Wisconsin Press, Madison, Wisconsin, 1983. Translated to English by Albert Biderman and Howard Wainer [*Cited on* pages 4, 13, 15, 29, 35, and 93].

E. Bertini and G. Santucci. Give chance a chance- modeling density to enhance scatter plot quality through random data sampling. *Information Visualization*, 5(2):95–110, 2006. DOI: 10.1057/palgrave.ivs.9500122 [*Cited on* pages 12, 30, 35, 60, and 64].

E. Bertini, A. Tatu, and D. Keim. Quality metrics in high-dimensional data visualization: An overview and systematization. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2203–2212, 2011. DOI: 10.1109/TVCG.2011.229 [*Cited on* pages 61 and 68].

L. Best, A. Hunter, and B. Stewart. Perceiving relationships: A physiological examination of the perception of scatterplots. *Diagrams*, pages 244–257, 2006. DOI: 10.1007/11783183_33 [*Cited on* pages 68 and 69].

M. Bostock, V. Ogievetsky, and J. Heer. D^3: Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011. DOI: 10.1109/TVCG.2011.185 [*Cited on* pages 72, 122, 127, 134, 136, and 145].

J. Boy, R. A. Rensink, E. Bertini, and J.-D. Fekete. A Principled Way of Assessing Visualization Literacy. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1963–1972, Dec 2014. DOI: 10.1109/TVCG.2014.2346984 [*Cited on* page 154].

M. Brehmer and T. Munzner. A Multi-Level Typology of Abstract Visualization Tasks. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2376–2385, Dec 2013. DOI: 10.1109/TVCG.2013.124 [*Cited on* pages 15, 16, 24, 31, 36, 62, 66, 67, and 76].

M. Brehmer, M. Sedlmair, S. Ingram, and T. Munzner. Visualizing dimensionally-reduced data: Interviews with Analysts and a Characterization of Task Sequences. In *Proc. Beyond Time and Errors Novel Evaluation Methods for Visualization (BELIV '14)*, pages 1–8, New York, New York, USA, 2014a. ACM Press. DOI: 10.1145/2669557.2669559 [*Cited on* pages 61, 64, 66, 67, and 70].

M. Brehmer et al. Overview: The design, adoption, and analysis of a visual document mining tool for investigative journalists. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2271–2280, 2014b. DOI: 10.1109/TVCG.2014.2346431 [*Cited on* page 54].

M. Brehmer et al. Matches, mismatches, and methods: Multiple-view workflows for energy portfolio analysis. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):449–458, 2016. DOI: 10.1109/TVCG.2015.2466971 [*Cited on* pages 47 and 48].

C. A. Brewer, G. W. Hatchard, and M. A. Harrower. Colorbrewer in print: a catalog of color schemes for maps. *Cartography and Geographic Information Science*, 30(1):5–32, 2003 [*Cited on* page 122].

E. T. Brown, J. Liu, C. E. Brodley, and R. Chang. Dis-Function: Learning distance functions interactively. In *IEEE Conference on Visual Analytics Science and Technology*, pages 83–92. IEEE, 2012. DOI: 10.1109/VAST.2012.6400486 [*Cited on* pages 61, 82, and 87].

J. R. Brubaker, F. Kivran-Swaine, L. Taber, and G. R. Hayes. Grief-stricken in a crowd: The language of bereavement and distress in social media. In *Sixth International AAAI Conference on Weblogs and Social Media*, pages 42–49, Dublin, Ireland, 2012. The AAAI Press [*Cited on* page 32].

M. Bruls, K. Huizing, and J. Van Wijk. Squarified treemaps. In *Data Visualization*, pages 33–42. Springer, 2000 [*Cited on* page 98].

I. Buck and T. Purcell. A toolkit for computation on GPUs. In *GPU Gems*. Addison Wesley, 2004. (Chapter 37) [*Cited on* pages 138 and 141].

D. Caragea, D. Cook, and V. G. Honavar. Gaining insights into support vector machine pattern classifiers using projection-based tour methods. In *Proc. ACM Knowledge Discovery and Data Mining*, pages 251–256. ACM, 2001. DOI: 10.1145/502512.502547 [*Cited on* page 94].

S. Card and J. Mackinlay. The structure of the information visualization design space. In *Proceedings of the Information Visualization Conference*, pages 92–99, 1997. DOI: 10.1109/INFVIS.1997.636792 [*Cited on* pages 12, 30, 35, and 149].

S. K. Card, J. D. Mackinlay, and B. Shneiderman. Readings in Information Visualization: Using Vision to Think. In S. K. Card, J. D. Mackinlay, and B. Shneiderman, editors, *Information Display*, The Morgan Kaufmann Series in Interactive Technologies, chapter 1, pages 1–34. Morgan Kaufmann, San Francisco, CA, USA, 1999 [*Cited on* pages 4 and 19].

D. B. Carr, R. J. Littlefield, W. L. Nicholson, and J. S. Littlefield. Scatterplot matrix techniques for large N. *Journal of the American Statistical Association*, 82(398):424, 1987. DOI: 10.2307/2289444 [*Cited on* pages 59, 60, 61, 75, and 84].

S. M. Casner. Task-analytic approach to the automated design of graphic presentations. *ACM Transactions on Graphics*, 10(2):111–151, Apr 1991. DOI: 10.1145/108360.108361 [*Cited on* pages 6, 14, 15, 36, 62, 64, 66, and 149].

Y.-H. Chan, C. D. Correa, and K.-L. Ma. Flow-based scatterplots for sensitivity analysis. In *IEEE Symposium on Visual Analytics Science and Technology*, pages 43–50. IEEE, 2010. DOI: 10.1109/VAST.2010.5652460 [*Cited on* pages 68, 70, and 82].

H. Chen, W. Chen, H. Mei, Z. Liu, K. Zhou, W. Gu, and K.-L. Ma. Visual abstraction and exploration of multi-class scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1683–1692, 2014. DOI: 10.1109/TVCG.2014.2346594 [*Cited on* pages 60, 76, and 81].

S. Chen et al. Interactive visual discovering of movement patterns from sparsely sampled geo-tagged social media data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):270–279, 2016. DOI: 10.1109/TVCG.2015.2467619 [*Cited on* pages vii, 39, and 40].

J. Choo, C. Lee, H. Kim, H. Lee, Z. Liu, R. Kannan, C. D. Stolper, J. Stasko, B. L. Drake, and H. Park. VisIRR: Visual analytics for information retrieval and recommendation with large-scale document data. *IEEE Conference on Visual Analytics Science and Technology*, 1 (C):243–244, 2015. DOI: 10.1109/VAST.2014.7042511 [*Cited on* page 75].

G. M. Cipriano, G. N. Philips, Jr, and M. Gleicher. Local functional descriptors for surface comparison based binding prediction. *BMC Bioinformatics*, 13(1):314, 2012. DOI: 10.1186/1471-2105-13-314 [*Cited on* page 106].

CLC bio. CLC Genomics Workbench. . Accessed: 2014-03-28 [*Cited on* page 114].

W. S. Cleveland. *The Elements of Graphing Data*. Wadsworth Advanced Books and Software, Monterey, CA, USA, 1985 [*Cited on* pages 12, 58, 64, 66, 73, and 86].

W. S. Cleveland and R. McGill. Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods. *Journal of the American Statistical Association*, 79:531–554, 1984a. DOI: 10.2307/2288400 [*Cited on* page 4].

W. S. Cleveland and R. McGill. The Many Faces of a Scatterplot. *Journal of the American Statistical Association*, 79(388):807–822, Dec 1984b. DOI: 10.2307/2288711 [*Cited on* pages 12, 75, and 87].

W. S. Cleveland, M. E. McGill, and R. McGill. The Shape Parameter of a Two-Variable Graph. *Journal of the American Statistical Association*, 83(402):289, Jun 1988. DOI: 10.2307/2288843 [*Cited on* page 62].

C. Collins, G. Penn, and S. Carpendale. Bubble Sets: Revealing set relations with isocontours over existing visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 15 (6):1009–1016, 2009. DOI: 10.1109/TVCG.2009.122 [*Cited on* pages 60 and 85].

M. Correll and M. Gleicher. What shakespeare taught us about text visualization. In *The 2nd Workshop on Interactive Visual Text Analytics*, 2012 [*Cited on* page 64].

M. Correll and M. Gleicher. Implicit Uncertainty Visualization: Aligning Perception and Statistics. *Proceedings of the 2015 Workshop on Visualization for Decision Making Under Uncertainty*, 2015 [*Cited on* pages 23 and 51].

M. Correll, D. Albers, S. Franconeri, and M. Gleicher. Comparing averages in time series data. *Prof. ACM Conference on Human Factors in Computing Systems*, page 1095, 2012. DOI: 10.1145/2207676.2208556 [*Cited on* pages 22, 93, and 98].

M. Correll, E. Alexander, D. Albers, A. Sarikaya, and M. Gleicher. Navigating reductionism and holism in evaluation. In *Proc. Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization - BELIV '14*, pages 23–26. ACM Press, 2014. DOI: 10.1145/2669557.2669577 [*Cited on* page 7].

M. Correll et al. LayerCake: a tool for the visual comparison of viral deep sequencing data. *Bioinformatics*, 31(July):btv407, 2015 [*Cited on* page 117].

J. Cottam, A. Lumsdaine, and P. Wang. Overplotting: Unified solutions under Abstract Rendering. *Proceedings - 2013 IEEE International Conference on Big Data, Big Data 2013*, pages 9–16, 2013. DOI: 10.1109/BigData.2013.6691712 [*Cited on* pages 64, 68, 82, and 85].

B. Craft and P. Cairns. Beyond Guidelines: What Can We Learn from the Visual Information Seeking Mantra? In *Ninth International Conference on Information Visualisation (IV'05)*, volume 2005, pages 110–118. IEEE, 2005. DOI: 10.1109/IV.2005.28 [*Cited on* pages 4, 7, 12, 13, and 153].

Q. Cui, M. O. Ward, E. A. Rundensteiner, and J. Yang. Measuring data abstraction quality in multiresolution visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):709–716, 2006. DOI: 10.1109/TVCG.2006.161 [*Cited on* pages 11, 27, and 60].

W. Cui et al. How hierarchical topics evolve in large text corpora. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2281–2290, 2014. DOI: 10.1109/TVCG.2014.2346433 [*Cited on* page 45].

J. Dai and J. Cheng. HMMEditor: a visual editing tool for profile hidden Markov model. *BMC Genomics*, 9(Suppl 1):S8, 2008. DOI: 10.1186/1471-2164-9-S1-S8 [*Cited on* page 95].

T. N. Dang and L. Wilkinson. ScagExplorer: Exploring scatterplots by their scagnostics. In *IEEE Pacific Visualization Symposium*, pages 73–80. IEEE, 2014. DOI: 10.1109/PacificVis.2014.42 [*Cited on* pages 61 and 68].

N. D. Dees et al. MuSiC: Identifying mutational significance in cancer genomes. *Genome Research*, 22(8):1589–1598, 2012. DOI: 10.1101/gr.134635.111 [*Cited on* page 116].

C. Demiralp et al. *invis*: Exploring high-dimensional RNA sequences from in vitro selection. In *Proc. IEEE Symp. Biological Data Visualization*, pages 1–8, 2013. DOI: 10.1109/BioVis.2013.6664340 [*Cited on* page 116].

J. Díaz, J. Petit, and M. Serna. A survey of graph layout problems. *ACM Computing Surveys*, 34(3):313–356, 2002. DOI: 10.1145/568522.568523 [*Cited on* page 118].

L. Ding et al. Somatic mutations affect key pathways in lung adenocarcinoma. *Nature*, 455 (7216):1069–1075, 2008. DOI: 10.1038/nature07423 [*Cited on* page 116].

C. Dunne and B. Shneiderman. Motif simplification: improving network visualization readability with fan, connector, and clique glyphs. In *Proc. Conf. Human Factors in Computing Systems*, pages 3247–3256. ACM Press, 2013. DOI: 10.1145/2470654.2466444 [*Cited on* page 118].

M. Elliott and R. Rensink. Interference in the Perception of Two-Population Scatterplots. *Journal of Vision*, 15(12):893, 2015. DOI: 10.1167/15.12.893 [*Cited on* pages 68 and 69].

G. Ellis and A. Dix. A Taxonomy of Clutter Reduction for Information Visualisation. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1216–1223, Nov 2007. DOI: 10.1109/TVCG.2007.70535 [*Cited on* pages 12, 30, 35, 60, 63, and 83].

N. Elmqvist and J. D. Fekete. Hierarchical aggregation for information visualization: Overview, techniques, and design guidelines. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):439–454, 2010. DOI: 10.1109/TVCG.2009.84 [*Cited on* pages 12, 23, 28, 35, 66, 73, and 86].

F. Enault, R. Fremez, E. Baranowski, and T. Faraut. Alvira: Comparative genomics of viral strains. *Bioinformatics*, 23(16):2178–2179, 2007. DOI: 10.1093/bioinformatics/btm293 [*Cited on* page 117].

J. A. Fails and D. R. Olsen, Jr. Interactive machine learning. In *Proc. ACM Intelligent User Interfaces*, pages 39–45. ACM, 2003. DOI: 10.1145/604045.604056 [*Cited on* page 95].

J.-D. Fekete and C. Plaisant. Interactive information visualization of a million items. In *IEEE Symposium on Information Visualization*, volume 2002, pages 117–124, 2002. DOI: 10.1109/INFVIS.2002.1173156 [*Cited on* pages 11, 27, and 60].

J. A. Ferstay, C. B. Nielsen, and T. Munzner. Variant View: Visualizing sequence variants in their gene context. *IEEE Trans. Vis. Comput. Graph.*, 19(12):2546–2555, 2013. DOI: 10.1109/TVCG.2013.214 [*Cited on* pages 116 and 117].

M. Fink, J. H. Haunert, J. Spoerhase, and A. Wolff. Selecting the aspect ratio of a scatter plot based on its delaunay triangulation. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2326–2335, 2013. DOI: 10.1109/TVCG.2013.187 [*Cited on* page 71].

M. Fiume et al. Savant: Genome browser for high-throughput sequencing data. *Bioinformatics*, 26(16):1938–1944, 2010. DOI: 10.1093/bioinformatics/btq332 [*Cited on* page 116].

M. Friendly and D. Denis. The early origins and development of the scatterplot. *Journal of the History of the Behavioral Sciences*, 41(2):103–130, 2005. DOI: 10.1002/jhbs.20078 [*Cited on* pages 58 and 66].

E. R. Gansner, Y. Hu, S. North, and C. Scheidegger. Multilevel agglomerative edge bundling for visualizing large graphs. In *Proceedings of the IEEE Pacific Visualization Symposium*, pages 187–194. IEEE, 2011. DOI: 10.1109/PACIFICVIS.2011.5742389 [*Cited on* page 139].

M. Ghoniem, J.-D. Fekete, and P. Castagliola. A comparison of the readability of graphs using node-link and matrix-based representations. In *IEEE Symp. Information Visualization*, pages 17–24. IEEE, 2004. DOI: 10.1109/INFVIS.2004.1 [*Cited on* page 117].

M. Gleicher, M. Correll, C. Nothelfer, and S. Franconeri. Perception of average value in multiclass scatterplots. *IEEE Transactions on Visualization and Computer Graphics*, 19(12): 2316–2325, 2013. DOI: 10.1109/TVCG.2013.183 [*Cited on* pages 22, 62, 64, 68, 69, 82, and 86].

M. Gleicher et al. Visual comparison for information visualization. *Information Visualization*, 10(4):289–309, Sept 2011. DOI: 10.1177/1473871611416549 [*Cited on* page 36].

D. Gotz and H. Stavropoulos. DecisionFlow: Visual analytics for high-dimensional temporal event sequence data. *IEEE Transactions on Visualization and Computer Graphics*, 20(12): 1783–1792, 2014. DOI: 10.1109/TVCG.2014.2346682 [*Cited on* page 117].

C. C. Gramazio, K. B. Schloss, and D. H. Laidlaw. The relation between visualization size, grouping, and user performance. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1953–1962, Dec 2014. DOI: 10.1109/TVCG.2014.2346983 [*Cited on* pages 68 and 69].

S. Gratzl et al. Domino: Extracting, comparing, and manipulating subsets across multiple tabular datasets. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2023–2032, 2014. DOI: 10.1109/TVCG.2014.2346260 [*Cited on* page 40].

H. Hagh-Shenas, S. Kim, V. Interrante, and C. Healey. Weaving versus blending: a quantitative assessment of the information carrying capacities of two alternative methods for conveying multivariate data with color. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1270–1277, 2007. DOI: 10.1109/TVCG.2007.70623 [*Cited on* page 5].

J. Halberda, S. F. Sires, and L. Feigenson. Multiple Spatially Overlapping Sets Can Be Enumerated in Parallel. *Psychological Science*, 17(7):572–576, Jul 2006. DOI: 10.1111/j.1467-9280.2006.01746.x [*Cited on* pages 22 and 93].

S. Haroz and D. Whitney. How Capacity Limits of Attention Influence Information Visualization Effectiveness. *IEEE Transactions on Visualization and Computer Graphics*, 18(12): 2402–2410, Dec 2012. DOI: 10.1109/TVCG.2012.233 [*Cited on* pages 13 and 22].

M. Harrower and C. Brewer. Colorbrewer.org: an online tool for selecting colour schemes for maps. *The Cartographic Journal*, 40(1):27–37, 2003. DOI: 10.1179/000870403235002042 [*Cited on* page 99].

C. G. Healey and J. T. Enns. Attention and Visual Memory in Visualization and Computer Graphics. *IEEE Transactions on Visualization and Computer Graphics*, 18(7):1170–1188, Jul 2012. DOI: 10.1109/TVCG.2011.127 [*Cited on* pages 22, 93, and 96].

C. G. Healey, K. S. Booth, and J. T. Enns. High-speed visual estimation using preattentive processing. *ACM Transactions on Computer-Human Interaction*, 3(2):107–135, 1996. DOI: 10.1145/230562.230563 [*Cited on* pages 62 and 64].

J. Heer and M. Agrawala. Multi-scale banking to 45°. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):701–708, 2006. DOI: 10.1109/TVCG.2006.163 [*Cited on* page 62].

J. Heer and B. Shneiderman. Interactive Dynamics for Visual Analysis. *Queue*, 10:30, 2012. DOI: 10.1145/2133416.2146416 [*Cited on* pages 15 and 36].

F. Heimerl, M. John, Q. Han, S. Koch, and T. Ertl. DocuCompass: Effective exploration of document landscapes. In *IEEE Conference on Visual Analytics Science and Technology*, pages 11–20, 2016. DOI: 10.1109/VAST.2016.7883507 [*Cited on* pages 83 and 86].

N. Henry and J.-D. Fekete. MatrixExplorer: A dual-representation system to explore social networks. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):677–684, 2006. DOI: 10.1109/TVCG.2006.160 [*Cited on* page 118].

K. Hornbæk and M. Hertzum. The notion of overview in information visualization. *International Journal of Human-Computer Studies*, 69(7-8):509–525, 2011. DOI: 10.1016/j.ijhcs.2011.02.007 [*Cited on* pages 10 and 93].

C. Hurter, O. Ersoy, and A. Telea. MoleView: An attribute and structure-based semantic lens for large element-based plots. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2600–2609, 2011. DOI: 10.1109/TVCG.2011.223 [*Cited on* page 77].

M. Imai et al. Experimental adaptation of an influenza H5 HA confers respiratory droplet transmission to a reassortant H5 HA/H1N1 virus in ferrets. *Nature*, 486(7403):420–428, 2012. DOI: 10.1038/nature10831 [*Cited on* pages 112 and 128].

A. Inselberg. Multidimensional detective. *Proc. VIZ '97: Visualization Conference, Information Visualization Symposium and Parallel Rendering Symposium*, pages 1–8, 1997. DOI: 10.1109/INFVIS.1997.636793 [*Cited on* page 118].

O. Irsoy, O. T. Yildiz, and E. Alpaydin. Design and analysis of classifier learning experiments in bioinformatics: Survey and case studies. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 9(6):1663–1675, Nov. 2012. DOI: 10.1109/TCBB.2012.117 [*Cited on* page 91].

W. Javed, B. McDonnel, and N. Elmqvist. Graphical perception of multiple time series. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):927–34, 2010. DOI: 10.1109/TVCG.2010.162 [*Cited on* pages 22 and 114].

Ji Soo Yi, Youn ah Kang, J. Stasko, and J. Jacko. Toward a Deeper Understanding of the Role of Interaction in Information Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1224–1231, Nov 2007. DOI: 10.1109/TVCG.2007.70515 [*Cited on* pages 16, 30, and 36].

I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, second edition, 2002. DOI: 10.1002/9781118445112.stat06472 [*Cited on* page 35].

E. Kandogan. Visualizing multi-dimensional clusters, trends, and outliers using star coordinates. In *ACM International Conference on Knowledge Discovery and Data Mining*, pages 107–116, New York, New York, USA, 2001. ACM Press. DOI: 10.1145/502512.502530 [*Cited on* pages 61 and 87].

A. Karve and M. Gleicher. Glyph-based overviews of large datasets in structural bioinformatics. In *IEEE 2007 Conf. Information Visualization (Supplements)*, pages 1–6, 2007 [*Cited on* page 94].

M. Kearse et al. Geneious Basic: An integrated and extendable desktop software platform for the organization and analysis of sequence data. *Bioinformatics*, 28(12):1647–1649, 2012. DOI: bioinformatics/bts199 [*Cited on* page 114].

J. Kehrer and H. Hauser. Visualization and visual analysis of multifaceted scientific data: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 19(3):495–513, 2013. DOI: 10.1109/TVCG.2012.110 [*Cited on* page 29].

D. A. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8, 2002. DOI: 10.1109/2945.981847 [*Cited on* page 28].

D. A. Keim and H. Kriegel. Visualization techniques for mining large databases: A comparison. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):923–938, 1996. DOI: 10.1109/69.553159 [*Cited on* page 28].

D. A. Keim, M. C. Hao, U. Dayal, H. Janetzko, and P. Bak. Generalized scatter plots. *Information Visualization*, 9(4):301–311, 2010. DOI: 10.1057/ivs.2009.34 [*Cited on* pages 60, 68, 76, 82, 84, and 85].

W. J. Kent et al. The Human Genome Browser at UCSC. *Genome Research*, pages 996–1006, 2002. DOI: 10.1101/gr.229102 [*Cited on* page 116].

N. A. Khazanov and H. A. Carlson. Exploring the composition of protein-ligand binding sites on a large scale. *PLoS Computational Biology*, 9(11):e1003321, 2013. DOI: 10.1371/journal.pcbi.1003321 [*Cited on* page 94].

Khronos Group. Typed array specification. , 2015a. D. Herman and K. Russell, editors. Accessed: 2015-07-20 [*Cited on* page 137].

Khronos Group. WebGL 1.0 specification. , 2015b. D. Jackson and J. Gilbert, editors. Accessed: 2015-07-20 [*Cited on* pages 136, 137, and 140].

H. Kim, J. Choo, H. Park, and A. Endert. InterAxis: Steering scatterplot axes via observation-level interaction. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):131–140, 2016. DOI: 10.1109/TVCG.2015.2467615 [*Cited on* pages 61, 77, and 87].

R. Kincaid and K. Dejgaard. MassVis: Visual analysis of protein complexes using mass spectrometry. In *IEEE Symposium on Visual Analytics Science and Technology*, pages 163–170, 2009. DOI: 10.1109/VAST.2009.5333895 [*Cited on* page 86].

G. Kindlmann and C. Scheidegger. An Algebraic Process for Visualization Design. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2181–2190, Dec 2014. DOI: 10.1109/TVCG.2014.2346325 [*Cited on* page 13].

G. Klein, B. Moon, and R. Hoffman. Making Sense of Sensemaking 2: A Macrocognitive Model. *IEEE Intelligent Systems*, 21(5):88–92, 2006. DOI: 10.1109/MIS.2006.100 [*Cited on* pages 14, 15, 20, and 36].

A. Kochtchi et al. Networks of Names: Visual exploration and semi-automatic tagging of social networks from newspaper articles. *Computer Graphics Forum*, 33(3):211–220, 2014. DOI: 10.1111/cgf.12377 [*Cited on* pages 43 and 49].

R. Kosara. Presentation-Oriented Visualization Techniques. *IEEE Computer Graphics and Applications*, 36(1):80–85, Jan 2016. DOI: 10.1109/MCG.2016.2 [*Cited on* pages 11, 29, 40, and 55].

B. Lee, C. Plaisant, C. S. Parr, J.-D. Fekete, and N. Henry. Task taxonomy for graph visualization. In *Proc. 2006 AVI BELIV*, pages 1–5, New York, New York, USA, 2006. ACM Press. DOI: 10.1145/1168149.1168168 [*Cited on* page 63].

D. J. Lehmann and H. Theisel. Orthographic star coordinates. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2615–2624, 2013. DOI: 10.1109/TVCG.2013.182 [*Cited on* pages 61 and 87].

D. J. Lehmann, F. Kemmler, T. Zhyhalava, M. Kirschke, and H. Theisel. Visualnostics: Visual guidance pictograms for analyzing projections of high-dimensional data. *Computer Graphics Forum*, 34(3):291–300, 2015. DOI: 10.1111/cgf.12641 [*Cited on* page 87].

Y. K. Leung and M. D. Apperley. E3: Towards the metrication of graphical presentation techniques for large data sets. In L. J. Bass, J. Gornostaev, and C. Unger, editors, *Proceedings of EWHCI '93*, pages 125–140. Springer, 1993. DOI: 10.1007/3-540-57433-6_44 [*Cited on* pages 13 and 29].

H. Li et al. The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, 25(16): 2078–2079, 2009 [*Cited on* page 127].

J. Li, J.-B. Martens, and J. J. van Wijk. Judging correlation from scatterplots and parallel coordinate plots. *Information Visualization*, 9(1):13–30, 2008. DOI: 10.1057/palgrave.ivs.9500179 [*Cited on* pages 62, 64, 66, 68, and 86].

J. Li, J.-B. Martens, and J. J. van Wijk. A model of symbol size discrimination in scatterplots. In *Proc. Conference on Human Factors in Computing Systems*, page 2553, New York, New York, USA, 2010a. ACM Press. DOI: 10.1145/1753326.1753714 [*Cited on* pages 62 and 86].

J. Li, J. J. van Wijk, and J.-B. Martens. A model of symbol lightness discrimination in sparse scatterplots. In *IEEE Pacific Visualization Symposium*, pages 105–112, 2010b. DOI: 10.1109/PACIFICVIS.2010.5429604 [*Cited on* pages 62, 64, 66, and 86].

Z. Liu and J. T. Stasko. Mental models, visual reasoning and interaction in information visualization: a top-down perspective. *IEEE transactions on visualization and computer graphics*, 16(6):999–1008, 2010. DOI: 10.1109/TVCG.2010.177 [*Cited on* pages 19 and 20].

Z. Liu, B. Jiang, and J. Heer. imMens: Real-time visual querying of big data. *Computer Graphics Forum*, 32(3pt4):421–430, 2013. DOI: 10.1111/cgf.12129 [*Cited on* pages 132 and 137].

A. M. MacEachren. *How Maps Work: Representation, Visualization, and Design*. The Guilford Press, New York, New York, USA, 1995 [*Cited on* pages 62 and 68].

J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Transactions on Graphics*, 5(2):110–141, 1986. DOI: 10.1145/22949.22950 [*Cited on* pages 63 and 86].

J. Mackinlay, P. Hanrahan, and C. Stolte. Show Me: Automatic Presentation for Visual Analysis. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1137–1144, Nov 2007. DOI: 10.1109/TVCG.2007.70594 [*Cited on* page 152].

E. Maguire et al. Redesigning the sequence logo with glyph-based approaches to aid interpretation. In *EuroVis—Short Papers*. The Eurographics Association, 2014. DOI: 10.2312/eurovisshort.20141159 [*Cited on* page 117].

J. Matejka and G. Fitzmaurice. Same Stats, Different Graphs: Generating Datasets with Varied Appearance and Identical Statistics through Simulated Annealing. In *Proc. Conference on Human Factors in Computing Systems*, pages 1290–1294, New York, New York, USA, 2017. ACM Press. DOI: 10.1145/3025453.3025912 [*Cited on* page 21].

J. Matejka, F. Anderson, and G. Fitzmaurice. Dynamic opacity optimization for scatter plots. In *Proc. Conference on Human Factors in Computing Systems*, pages 2707–2710, New York, New York, USA, 2015. ACM Press. DOI: 10.1145/2702123.2702585 [*Cited on* page 85].

B. W. Matthews. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta*, 405(2):442–451, 1975. DOI: 10.1016/0005-2795(75)90109-9 [*Cited on* page 95].

A. Mayorga and M. Gleicher. Splatterplots: Overcoming overdraw in scatter plots. *IEEE Transactions on Visualization and Computer Graphics*, 19(9):1526–1538, 2013. DOI: 10.1109/TVCG.2013.65 [*Cited on* pages 43, 44, 59, 60, 68, 75, 81, 82, 85, 135, 138, and 140].

B. McDonnel and N. Elmqvist. Towards utilizing GPUs in information visualization: a model and implementation of image-space operations. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1105–12, 2009. DOI: 10.1109/TVCG.2009.191 [*Cited on* pages 122 and 136].

D. R. Montello, S. I. Fabrikant, M. Ruocco, and R. S. Middleton. Testing the first law of cognitive geography on point-display spatializations. *Cosit*, pages 316–331, 2003. DOI: 10.1007/978-3-540-39923-0_21 [*Cited on* pages 62, 66, 68, and 70].

D. Moritz, J. Heer, and B. Howe. Dynamic Client-Server Optimization for Scalable Interactive Visualization on the Web. In *Workshop on Data Systems for Interactive Analysis (DSIA '15)*, 2015 [*Cited on* page 132].

Mozilla Developer Network. DataView — JavaScript. , 2015. Accessed: 2015-07-20 [*Cited on* page 144].

P. M. Mullins and S. Treu. A task-based cognitive model for user-network interaction: defining a task taxonomy to guide the interface designer. *Interacting with Computers*, 5(2): 139–166, 1993. DOI: http://dx.doi.org/10.1016/0953-5438(93)90016-M [*Cited on* page 14].

T. Munzner. A nested model for visualization design and validation. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):921–928, 2009. DOI: 10.1109/TVCG.2009.111 [*Cited on* pages 15 and 23].

T. Munzner. *Visualization Analysis & Design*. CRC Press: Taylor & Francis Group, Boca Raton, FL, 2014 [*Cited on* pages 2, 16, 57, 58, 62, 66, and 152].

C. B. Nielsen et al. Visualizing genomes: techniques and challenges. *Nature Methods*, 7(3): S5–S15, 2010. DOI: 10.1038/nmeth.1422 [*Cited on* page 116].

D. Norman. *The Pyschology of Everyday Things*. Basic Books, New York, 1988 [*Cited on* page 18].

S. O'Connor et al. Conditional CD8+ T cell escape during acute simian immunodeficiency virus infection. *Journal of Virology*, 86(1):605–609, 2012. DOI: 10.1128/JVI.05511-11 [*Cited on* page 130].

S. I. O'Donoghue, D. S. Goodsell, A. S. Frangakis, F. Jossinet, R. A. Laskowski, M. Nilges, H. R. Saibil, A. Schafferhans, R. C. Wade, E. Westhof, et al. Visualization of macromolecular structures. *Nature Methods*, 7:S42–S55, 2010. DOI: 10.1038/nmeth.1427 [*Cited on* page 94].

S. E. Palmer. Common region: A new principle of perceptual grouping. *Cognitive Psychology*, 24(3):436–447, 1992. DOI: 10.1016/0010-0285(92)90014-S [*Cited on* page 99].

K. Perlin. An image synthesizer. volume 19, pages 287–296. ACM, July 1985. DOI: 10.1145/325165.325247 [*Cited on* page 103].

W. A. Pike, J. Stasko, R. Chang, and T. A. O'Connell. The science of interaction. *Information Visualization*, 8(4):263–274, 2009. DOI: 10.1057/ivs.2009.22 [*Cited on* pages 16 and 31].

P. Pirolli and S. Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of International Conference on Intelligence Analysis*, 2005 [*Cited on* pages 18 and 19].

T. Polk et al. Tennivis: Visualization for tennis match analysis. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2339–2348, 2014. DOI: 10.1109/TVCG.2014.2346445 [*Cited on* page 53].

K. Potter, J. Kniss, R. Riesenfeld, and C. Johnson. Visualizing Summary Statistics and Uncertainty. *Computer Graphics Forum*, 29(3):823–832, Aug 2010. DOI: 10.1111/j.1467-8659.2009.01677.x [*Cited on* page 20].

D. Powers. Evaluation: From precision, recall and f-measure to ROC, informedness, markedness & correlation. *Journal of Machine Learning Techology*, 2(1):37–63, 2011 [*Cited on* page 95].

R. Rensink and G. Baldridge. The perception of correlation in scatterplots. *Computer Graphics Forum*, 29(3):1203–1210, 2010. DOI: 10.1111/j.1467-8659.2009.01694.x [*Cited on* pages 62, 64, 66, and 68].

R. A. Rensink. The nature of correlation perception in scatterplots. *Psychonomic Bulletin & Review*, 24(3):776–797, 2017. DOI: 10.3758/s13423-016-1174-7 [*Cited on* page 71].

D. Riffe et al. *Analyzing Media Messages: Using Quantitative Content Analysis in Research*. Lawrence Erlbaum Associates, Inc., Mahwah, NJ, USA, 1998 [*Cited on* pages 26 and 32].

A. Rind, W. Aigner, M. Wagner, S. Miksch, and T. Lammarsch. Task Cube: A three-dimensional conceptual space of user tasks in visualization design and evaluation. *Information Visualization*, 15(4):288–300, Oct 2016. DOI: 10.1177/1473871615621602 [*Cited on* pages 4, 16, 31, and 62].

J. T. Robinson et al. Integrative genomics viewer. *Nature Biotechnology*, 29(1):24–26, 2011. DOI: 10.1038/nbt.1754 [*Cited on* page 114].

G. Rong and T.-S. Tan. Jump flooding in GPU with applications to voronoi diagram and distance transform. In *Proceedings of the Symposium on Interactive 3D Graphics and Games*, pages 109–116. ACM, 2006 [*Cited on* page 138].

R. E. Roth. An Empirically-Derived Taxonomy of Interaction Primitives for Interactive Cartography and Geovisualization. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2356–2365, Dec 2013a. DOI: 10.1109/TVCG.2013.130 [*Cited on* page 64].

R. E. Roth. Interactive maps: What we know and what we need to know. *Journal of Spatial Information Science*, 6(6):59–115, Jun 2013b. DOI: 10.5311/JOSIS.2013.6.105 [*Cited on* page 24].

S. F. Roth and J. Mattis. Data characterization for intelligent graphics presentation. *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 193–200, 1990. DOI: 10.1145/97243.97273 [*Cited on* pages 14 and 36].

B. Saket, H. Kim, E. T. Brown, and A. Endert. Visualization by demonstration: An interaction paradigm for visual data exploration. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):331–340, 2017. DOI: 10.1109/TVCG.2016.2598839 [*Cited on* page 80].

R. Sanjuán et al. Viral mutation rates. *Journal of Virology*, 84(19):9733–9748, 2010. DOI: 10.1128/JVI.00694-10 [*Cited on* page 111].

M. F. Sanner, A. J. Olson, and J.-C. Spehner. Reduced surface: An efficient way to compute molecular surfaces. *Biopolymers*, 38(3):305–320, 1996. DOI: 10.1002/(SICI)1097-0282(199603)38:3<305::AID-BIP4>3.0.CO;2-Y [*Cited on* pages 99 and 104].

P. Saraiya, C. North, and K. Duca. An insight-based methodology for evaluating bioinformatics visualizations. *IEEE transactions on visualization and computer graphics*, 11(4):443–56, 2005. DOI: 10.1109/TVCG.2005.53 [*Cited on* page 19].

P. Saraiya, C. North, and K. Duca. An Insight-Based Longitudinal Study of Visual Analytics. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1511–1522, 2006. DOI: 10.1109/TVCG.2006.85 [*Cited on* pages 19 and 53].

A. Sarikaya and M. Gleicher. Scatterplots: Tasks, data, and designs. *IEEE Transactions on Visualization and Computer Graphics*, 24, 2018. (to appear) [*Cited on* pages 57, 72, and 83].

A. Sarikaya, D. Albers, J. Mitchell, and M. Gleicher. Visualizing validation of protein surface classifiers. *Computer Graphics Forum*, 33(3):171–180, Jun 2014. DOI: 10.1111/cgf.12373 [*Cited on* pages 5 and 90].

A. Sarikaya, M. Correll, J. M. Dinis, D. H. O'Connor, and M. Gleicher. Visualizing Co-occurrence of Events in Populations of Viral Genome Sequences. *Computer Graphics Forum*, 35(3):151–160, 2016 [*Cited on* page 110].

A. Satyanarayan and J. Heer. Lyra: An interactive visualization design environment. *Computer Graphics Forum*, 33(3):351–360, Jun 2014. DOI: 10.1111/cgf.12391 [*Cited on* page 153].

C. Scheidegger. encodeFloat() — Lux. , 2015. Accessed: 2015-09-15 [*Cited on* page 141].

H. J. Schulz, T. Nocke, M. Heitzler, and H. Schumann. A design space of visualization tasks. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2366–2375, 2013. DOI: 10.1109/TVCG.2013.120 [*Cited on* pages 15, 16, 24, 29, 31, 35, 36, 62, 63, 66, 67, and 77].

D. W. Scott. Kernel density estimators. In *Multivariate Density Estimation*, pages 125–193. John Wiley & Sons, Inc., 2008. DOI: 10.1002/9780470316849.ch6 [*Cited on* pages 81 and 84].

M. Sedlmair, M. Meyer, and T. Munzner. Design study methodology: Reflections from the trenches and the stacks. *IEEE Transactions on Visualization and Computer Graphics*, 18(12): 2431–2440, Dec 2012a. DOI: 10.1109/TVCG.2012.213 [*Cited on* pages 1, 88, 89, 111, 149, and 150].

M. Sedlmair, A. Tatu, T. Munzner, and M. Tory. A taxonomy of visual cluster separation factors. *Computer Graphics Forum*, 31(3pt4):1335–1344, 2012b. DOI: 10.1111/j.1467-8659.2012.03125.x [*Cited on* pages 63 and 68].

M. Sedlmair, T. Munzner, and M. Tory. Empirical guidance on scatterplot and dimension reduction technique choices. *IEEE Transactions on Visualization and Computer Graphics*, 19 (12):2634–2643, 2013. DOI: 10.1109/TVCG.2013.153 [*Cited on* pages 63, 64, 66, 68, 70, and 145].

J. Serra. *Image analysis and mathematical morphology*. London: Academic Press, 1982 [*Cited on* page 100].

L. Shi et al. SAVE: Sensor anomaly visualization engine. In *IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 201–210. IEEE, 2011. DOI: 10.1109/VAST.2011.6102458 [*Cited on* page 49].

B. Shneiderman. The eyes have it: a task by data type taxonomy for information visualizations. In *Proceedings IEEE Symposium on Visual Languages*, pages 336–343. IEEE Comput. Soc. Press, 1996. DOI: 10.1109/VL.1996.545307 [*Cited on* pages 4, 12, 27, 30, 35, 44, and 53].

B. Shneiderman and C. Plaisant. Sharpening Analytic Focus to Cope with Big Data Volume and Variety. *IEEE Computer Graphics and Applications*, 35(3):10–14, May 2015. DOI: 10.1109/MCG.2015.64 [*Cited on* pages 20 and 124].

M. Sips, B. Neubert, J. P. Lewis, and P. Hanrahan. Selecting good views of high-dimensional data using class consistency. *Computer Graphics Forum*, 28(3):831–838, 2009. DOI: 10.1111/j.1467-8659.2009.01467.x [*Cited on* pages 64, 66, 68, 69, and 70].

A. Slingsby, J. Dykes, and J. Wood. Configuring hierarchical layouts to address research questions. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):977–84, 2009. DOI: 10.1109/TVCG.2009.128 [*Cited on* pages 13 and 93].

R. Spence. *Information Visualization: Design for Interaction*. Pearson Education Limited, Harlow, England, 2nd edition, 2007 [*Cited on* pages 10 and 19].

D. Spencer. *Card Sorting: Designing Usable Categories*. Rosenfield Media, Brooklyn, New York, 2009 [*Cited on* page 64].

R. R. Springmeyer, M. M. Blattner, and N. L. Max. A characterization of the scientific data analysis process. In *Proceedings of the IEEE Conference on Visualization*, pages 235–242, 1992 [*Cited on* pages 13 and 36].

J. Staib, S. Grottel, and S. Gumhold. Enhancing Scatterplots with Multi-Dimensional Focal Blur. *Computer Graphics Forum*, 35(3):11–20, 2016. DOI: 10.1111/cgf.12877 [*Cited on* pages 85 and 86].

S. V. Stehman. Selecting and interpreting measures of thematic classification accuracy. *Remote Sensing of Environment*, 62(1):77–89, 1997. DOI: 10.1016/S0034-4257(97)00083-7 [*Cited on* page 95].

R. Steuer et al. The mutual information: detecting and evaluating dependencies between variables. *Bioinformatics (Oxford, England)*, 18 Suppl 2:S231–S240, 2002 [*Cited on* pages 114 and 120].

C. D. Stolper et al. Progressive visual analytics: User-driven visual exploration of in-progress analytics. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1653–1662, 2014. DOI: 10.1109/TVCG.2014.2346574 [*Cited on* page 45].

M. Stone, D. A. Szafir, and V. Setlur. An engineering model for color difference as a function of size. In *IS&T 22nd Color Imaging Conference*, pages 253–258, 2014 [*Cited on* page 86].

H. Strobelt, B. Alsallakh, J. Botros, B. Peterson, M. Borowsky, H. Pfister, and A. Lex. Vials: Visualizing Alternative Splicing of Genes. *IEEE Trans. Vis. Comput. Graph.*, 22(1):399–408, 2016. DOI: 10.1109/TVCG.2015.2467911 [*Cited on* pages 116 and 117].

D. A. Szafir, S. Haroz, M. Gleicher, and S. Franconeri. Four types of ensemble coding in data visualizations. *Journal of Vision*, 16(5):11, Mar 2016. DOI: 10.1167/16.5.11 [*Cited on* pages 23 and 52].

J. Talbot, B. Lee, A. Kapoor, and D. S. Tan. EnsembleMatrix: Interactive visualization to support machine learning with multiple classifiers. In *Proc. 2009 ACM Human Factors in Computing Systems*, pages 1283–1292. ACM, 2009. DOI: 10.1145/1518701.1518895 [*Cited on* page 95].

J. Talbot, J. Gerth, and P. Hanrahan. Arc length-based aspect ratio selection. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2276–2282, Dec 2011. DOI: 10.1109/TVCG.2011.167 [*Cited on* page 62].

M. Tarini, P. Cignoni, and C. Montani. Ambient occlusion and edge cueing for enhancing real time molecular visualization. *IEEE Transactions of Visualization and Computer Graphics*, 12(5):1237–1244, 2006. DOI: 10.1109/TVCG.2006.115 [*Cited on* page 99].

A. Tatu, P. Bak, E. Bertini, D. Keim, and J. Schneidewind. Visual quality metrics and human perception. In *Proceedings of the International Conference on Advanced Visual Interfaces*, pages 49–56, New York, New York, USA, 2010. ACM Press. DOI: 10.1145/1842993.1843002 [*Cited on* pages 61, 64, 66, and 68].

A. Thudt, U. Hinrichs, and S. Carpendale. The bohemian bookshelf: supporting serendipitous book discoveries through information visualization. In *ACM Conference on Human Factors in Computing Systems*, pages 1461–1470, New York, New York, USA, 2012a. ACM Press. DOI: 10.1145/2207676.2208607 [*Cited on* pages 6 and 11].

A. Thudt et al. The Bohemian Bookshelf: Supporting serendipitous book discoveries through information visualization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1461–1470. ACM, 2012b. DOI: 10.1145/2207676.2208607 [*Cited on* page 52].

C. Tominski, S. Gladisch, U. Kister, R. Dachselt, and H. Schumann. Interactive lenses for visualization: An extended survey. *Computer Graphics Forum (STAR)*, (00):1–28, 2016. DOI: 10.1111/cgf.12871 [*Cited on* pages 82 and 83].

M. Tory, D. Sprague, F. Wu, W. Y. So, and T. Munzner. Spatialization design: comparing points and landscapes. *IEEE Transactions on Visualization and Computer Graphics*, 13(6): 1262–1269, 2007. DOI: 10.1109/TVCG.2007.70596 [*Cited on* pages 60, 64, 68, 81, and 82].

B. E. Trumbo. A theory for coloring bivariate statistical maps. *The American Statistician*, 35 (4):220–226, 1981. DOI: 10.1080/00031305.1981.10479360 [*Cited on* page 122].

Tuan Nhon Dang, L. Wilkinson, and A. Anand. Stacking graphic elements to avoid overplotting. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1044–1052, 2010. DOI: 10.1109/TVCG.2010.197 [*Cited on* page 64].

E. R. Tufte. *Envisioning information*. Graphics Press LLC, Chreshire, CT, 1990 [*Cited on* page 93].

T. Urness, V. Interrante, I. Marusic, E. Longmire, and B. Ganapathisubramani. Effectively visualizing multi-valued flow data using color and texture. In *IEEE Visualization Con-*

ference, pages 115–121. IEEE, 2003. DOI: 10.1109/VISUAL.2003.1250362 [*Cited on* pages 5 and 22].

D. K. Urribarri and S. M. Castro. Prediction of data visibility in two-dimensional scatterplots. *Information Visualization*, 16(2):113–125, 2017. DOI: 10.1177/1473871616638892 [*Cited on* pages 68 and 69].

R. B. J. Van Brakel et al. COMBat: Visualizing co-occurrence of annotation terms. In *Proc. IEEE Symp. Biological Data Visualization*, pages 17–24. IEEE, 2013. DOI: 10.1109/BioVis.2013.6664342 [*Cited on* pages 116 and 117].

S. van den Elzen and J. J. van Wijk. BaobabView: Interactive construction and analysis of decision trees. In *2011 IEEE Conf. Visual Analytics Science and Technology*, pages 151–160, 2011. DOI: 10.1109/VAST.2011.6102453 [*Cited on* page 94].

J. van Wijk. The Value of Visualization. In *IEEE Visualization Conference*, volume 12, pages 79–86. IEEE, 2005. DOI: 10.1109/VISUAL.2005.1532781 [*Cited on* page 13].

P.-P. Vázquez, M. Feixas, M. Sbert, and W. Heidrich. Viewpoint selection using viewpoint entropy. In *Proc. Vision, Modeling and Visualization Conference*, volume 1010, pages 273–280, 2001 [*Cited on* page 101].

C. Ware. Quantitative texton sequences for legible bivariate maps. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1523–1530, 2009. DOI: 10.1109/TVCG.2009.175 [*Cited on* pages 102 and 123].

C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann, Burlington, MA, 3rd edition, 2012 [*Cited on* pages 21, 73, 75, 86, 93, and 102].

J. Waser et al. World lines. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1458–1467, 2010. DOI: 10.1109/TVCG.2010.223 [*Cited on* page 41].

M. Wertheimer. Untersuchungen zur Lehre von der Gestalt II [in English: "Laws of Organization in Perceptual Forms"]. *Psycologische Forschung*, 4:301–350, 1923. Translated to English by W. Ellis, appearing in *A Source Book of Gestalt Psychology*, pp. 71-88, 1938. [*Cited on* page 21].

P. R. Wilker, J. M. Dinis, et al. Selection on haemagglutinin imposes a bottleneck during mammalian transmission of reassortant H5N1 influenza viruses. *Nature Communications*, 4:2636, 2013 [*Cited on* pages 128 and 129].

L. Wilkinson. *The Grammar of Graphics (Statistics and Computing)*. Spinger Science+Business Media, Inc., New York, New York, USA, 2nd edition, 2005 [*Cited on* page 73].

L. Wilkinson, A. Anand, and R. Grossman. Graph-theoretic scagnostics. In *IEEE Symposium on Information Visualization*, pages 157–164. IEEE, 2005. DOI: 10.1109/INFVIS.2005.1532142 [*Cited on* pages 61, 68, and 70].

R. Williams. *The Non-Designer's Design Book: Design and Typographic Principles for the Visual Novice*. Peachpit Press, San Francisco, California, 2015 [*Cited on* page 150].

I. H. Witten, E. Frank, and M. A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques*. Elsevier, 2011 [*Cited on* page 95].

A. Yates, A. Webb, M. Sharpnack, H. Chamberlin, K. Huang, and R. Machiraju. Visualizing multidimensional data with glyph SPLOMs. *Computer Graphics Forum*, 33(3):301–310, 2014. DOI: 10.1111/cgf.12386 [*Cited on* pages 47, 48, 61, 75, and 87].

J. S. Yi, Y.-a. Kang, J. T. Stasko, and J. a. Jacko. Understanding and characterizing insights. In *Proceedings of the Conference on BEyond time and errors novel evaLuation methods for Information Visualization - BELIV '08*, page 1, New York, New York, USA, 2008. ACM Press. DOI: 10.1145/1377966.1377971 [*Cited on* page 20].

X. Yuan, D. Ren, Z. Wang, and C. Guo. Dimension Projection Matrix/Tree: Interactive subspace visual exploration and analysis of high dimensional data. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2625–2633, 2013. DOI: 10.1109/TVCG.2013.150 [*Cited on* page 61].

M. Zhou and S. Feiner. Visual task characterization for automated visual discourse synthesis. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 392–399, 1998. DOI: 10.1145/274644.274698 [*Cited on* pages 14, 30, and 36].

X. Zhu, S. S. Ericksen, and J. C. Mitchell. DBSI: DNA-binding site identifier. *Nucleic Acids Research*, 41(16):e160, 2013. DOI: 10.1093/nar/gkt617 [*Cited on* page 105].

C. Ziemkiewicz and R. Kosara. Laws of attraction: From perceptual forces to conceptual similarity. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1009–1016, 2010. DOI: 10.1109/TVCG.2010.174 [*Cited on* page 75].

C. Ziemkiewicz, A. Ottley, R. J. Crouser, K. Chauncey, S. L. Su, and R. Chang. Understanding Visualization by Understanding Individual Users. *IEEE Computer Graphics and Applications*, 32(6):88–94, Nov 2012. DOI: 10.1109/MCG.2012.120 [*Cited on* page 19].