

You can't always sketch what you want: Understanding Sensemaking in Visual Query Systems

Submission ID: 1051

Abstract

Visual query systems (VQSs) allow users to interactively search for line charts with desired visual patterns typically specified using intuitive sketch-based interfaces. Despite their potential in accelerating data exploration, more than a decade of past work on VQSs has not been translated to adoption in practice. Through a year-long collaboration with experts from three diverse domains, we examine the role of VQSs in real data exploration workflows, develop features to support these workflows by enhancing an existing VQS, and evaluate how these features are used in practice. Via these observations, we formalize a taxonomy of key functionalities for VQSs, organized by three sensemaking processes. Perhaps somewhat surprisingly, we find that ad-hoc sketch-based querying is not commonly used during data exploration, since analysts are often unable to precisely articulate the patterns they are interested in. We find that there is a spectrum of VQS-centric data exploration workflows, depending on the application, and that many of these workflows are not effectively supported in present-day VQSs. Our insights can pave the way for next-generation VQSs to be adopted in a variety of real-world applications.

Keywords: Visual analytics, exploratory analysis, visual query

1. Introduction

Line charts are commonly employed during data exploration—the intuitive connected patterns often illustrate complex underlying processes and yield interpretable and visually compelling data-driven narratives. To discover patterns in line charts, analysts construct them using toolkits like `ggplot` or `matplotlib`, or visualization construction interfaces like Excel or Tableau, specifying *exactly* what they want to visualize. For example, when trying to find celestial objects corresponding to supernovae, which have a specific pattern of brightness over time, astronomers individually inspect the corresponding line chart for each object (often numbering in the hundreds) until they find ones that match the pattern. This process of manual exploration of large numbers of line charts is not only error-prone, but also overwhelming for analysts.

To address this challenge, there has been a large number of papers dedicated to building *Visual Query Systems* (VQSs), that allow users to specify desired visual patterns via an interactive interface [MVK*11, HS04, Wat01, SKL*16, RLL*05, CG16, MA18, EZ15, HF09]. This interactive interface is one with a sketching canvas where users can draw a pattern of interest, with the system automatically traversing all potential visualization candidates to find those that match the specification. Since the intent of a sketch can be ambiguous, some work has developed mechanisms to enable users to clarify how a sketch should be interpreted [RLL*05, CG16, MA18, EZ15, HF09].

While this intuitive specification interface seems to be a promising solution to the problem of painful manual exploration of visualizations, to the best of our knowledge, VQSs are not very commonly used in practice. *Our paper seeks to bridge this gap to understand*

how VQSs can actually be used in practice, as a first step towards the broad adoption of VQSs in data analysis. Unlike prior work on VQSs, we set out to not only evaluate VQSs in-situ on real problem domains, but also involve participants from these domains in the VQS design. We present findings from a series of interviews, **contextual inquiry**, participatory design, and user studies with scientists from three different domains—*astronomy*, *genetics*, and *material science*—over the course of a year-long collaboration. These domains were selected to capture a diverse set of goals and datasets wherein VQSs can help address important scientific questions, such as: How does a treatment affect the expression of a gene in a breast cancer cell-line? Which battery components have sustainable levels of energy-efficiency and are safe and cheap to manufacture in production?

Via cognitive walkthroughs and interviews, we first identified challenges in existing data analysis workflows in these domains that could be potentially addressed by a VQS. Building on top of an existing, open-source VQS, *zenvisage* [SLK*17, SKL*16], we collaborated closely with our participants to gather feedback and iterate on VQS feature designs, over the course of a year, culminating in a new enhanced VQS, *zenvisage++*. We organized these features into a taxonomy of VQS functionalities, involving three sensemaking processes inspired by Pirolli and Card's notional model of analyst sensemaking [PC05]. The sensemaking processes include top-down pattern specification (translating a pattern “in-the-head” into the form of a visual query), bottom-up data-driven inquiries (querying or recommending based on data), and context-creation (navigating across different collections of visualizations). We find that prior VQSs have focused largely on top-down processes, while largely ignoring the other two processes that are crucial for the needs in all three domains.

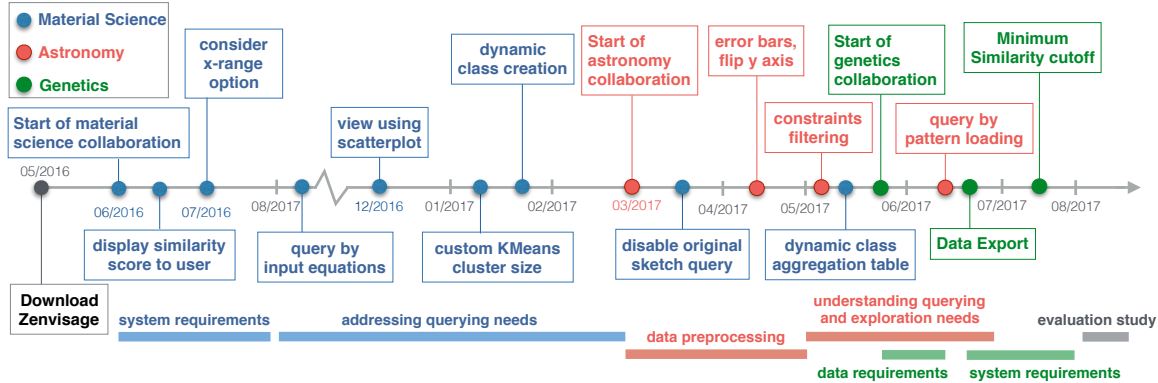


Figure 1: Timeline for progress in participatory design studies.

To study how various VQS features are used in practice, we conducted a final evaluation study with nine participants using our final VQS prototype, *zenvisage++*, to address their research questions on their own datasets. In a 1.5-hour user study, participants were able to gain novel scientific insights, such as identifying a star with a transient pattern that was known to harbor a Jupiter-sized planet and finding characteristic gene expression profiles confirming the results of a related publication.

By analyzing the evaluation study results, we discovered that sketching a pattern for querying is often ineffective. This is due to the fact that sketching makes the problematic assumptions that users know the pattern that they want to sketch and are able to sketch it precisely. Instead, participants typically opted for other means of pattern specification—one common mechanism was to drag-and-drop a recommended pattern onto the canvas, and then modify it (e.g., by smoothing it out). However, most VQSs do not support these other mechanisms (as we argued earlier, they typically focus only on top-down sensemaking processes, without covering bottom-up and context creation), partially explaining why such systems have not been widely adopted in practice.

Further analysis of how participants transition between different sensemaking processes during analysis—including the construction of a Markov Model—illustrated how participants adopt a diverse set of workflows tailored to their domains. We find that participants often construct analysis workflows focused around a primary sensemaking process, while iteratively interleaving their analysis with the two other processes. This finding points to how all three sensemaking processes, along with seamless transitions between them, are essential for enabling users to effectively use VQSs for data exploration.

To the best of our knowledge, our study is the *first to holistically examine how VQSs can be designed to fit the needs of real-world analysts and how they are actually used in practice*. Our contributions include:

- a characterization of the problems addressable by VQSs through design studies with three different domains,
- the construction of a taxonomy of functionalities within VQSs, as well as an articulation of the problem space that is amenable to VQSs, both grounded in participatory design findings,
- an integrative VQS, *zenvisage++*, capable of facilitating rapid hypothesis generation and insight discovery,

- study findings on how VQSs are used in practice, leading to the development of a novel sensemaking model for VQSs.

Our work not only opens up a new space of opportunities beyond the narrow use cases considered by prior studies, but also advocates common design guidelines and end-user considerations for building next-generation VQSs.

2. Methods

2.1. Background and Motivation

Visual query systems enable users to directly search for visualizations matching certain patterns through an intuitive specification interface. Early work in this space focused on interfaces to search for time series with specific patterns, including TimeSearcher [HS01, HS04], where the query specification mechanism is a rectangular box, filtering out all of the time series that does not pass through it, QuerySketch [Wat01] and Google Correlate [MVK*11], where the query is sketched as a pattern on canvas, filtering out all of the time series that have a different shape. Subsequent work recognized the ambiguity in sketching by studying how humans rank the similarity in patterns [EZ15, CG16, MA18] and improving the expressiveness of sketched queries through finer-grained specification interfaces and pattern-matching algorithms [RLL*05, HF09].

While these systems have been effective in controlled lab studies, they have never been designed and evaluated in-situ on real-world use cases. Even when use cases were involved [HS04, CG16], the inclusion of these use cases had a narrow objective and had little influence on the major design decisions of the system. In the context of Munzner's nested model [Mun09], this represents the common "downstream threat" of jumping prematurely into the deep levels of *encoding*, *interaction*, or *algorithm design*, before a proper *domain problem characterization* and *data/operation abstraction design*. In this work, we performed design studies [LBI*12, SP06, MS12] with three different subject areas for *domain problem characterization*. Comparing and contrasting between the diverse set of questions, datasets, and challenges across these three use cases revealed new generalizable insights and enabled us to better understand how VQSs can be extended for novel and unforeseen use cases. Based on these findings, we develop a feature taxonomy for understanding the sensemaking process in VQSs as part of the *data/operation abstraction design*. Finally, we validated the abstraction design with grounded evaluation [Pla04, IZCC08], where we invite participants

to bring in their own datasets and research problems that they have a vested interest in to test our final deployed system. Next, we will describe these two phases of our study in more detail.

2.2. Phase I: Participatory Design

We recruited participants by reaching out to research groups via email and word of mouth, who have experienced challenges in data exploration. Based on our early conversations with analysts from 12 different potential application areas, we narrowed down to three use cases in astronomy, genetics, and material science for our participatory design study, chosen based on their suitability for VQSs as well as diversity in use cases. **Six scientists, with an average of more than 6 years of research experience in their respective fields, participated in the design process.** Via interviews and contextual inquiries, we identified the needs and challenges of these use cases based on participant's original analysis workflow.

For the participatory design study, we built on an existing VQS, *zenvisage* [SLK*17, SKL*16], that allowed users to sketch a pattern or drag-and-drop an existing visualization as a query, with the system returning visualizations that had the closest Euclidean distance from the queried pattern. We chose to build on top of *zenvisage*, since it was open-source, extensible, and encompassed a large selection of features compared to existing systems, which focused largely on features for pattern and match specification (as compared in Table ??). Past research on participatory design have found that the use of functional prototypes is a common and effective way of engaging with participant and provide a starting point for participatory design [CAM*16]. Our motivation for providing a functional prototype at the beginning of the participatory design sessions is to showcase capabilities of VQSs. Especially since VQSs are not common in the existing workflows of these scientists, participants may not be able to imagine their use cases without a starting point.

During participatory design, we collaborated with each team closely with an average of two meetings per month, where we learned about their datasets, objectives, and **what additional VQS functionalities** could help address their research questions. Since some of the essential features that were crucial for effective exploration were lacking in *zenvisage* and still under development in the new version of our VQS, *zenvisage++*, we did not provide a deployed prototype for participants to actively use on their own during the participatory design period. Instead, as we iterated on the design of these features, relevant functionalities from intermediate versions of *zenvisage++* were demonstrated to the participants to elicit feedback on how VQSs could better support their scientific goals. The use of 'simulated future work situation' is common in cooperative prototyping when the real use of the prototype is not feasible [SG91]. A summary timeline of our collaboration with participants over a year and features inspired by their use cases can be found in Figure 1. Through this process, we identified and incorporated more than 20 desired features into *zenvisage++*. **Given the space limitation of our paper, we will focus our discussion in Section 4 on the major *zenvisage++* functionalities relevant to the study findings, and defer the details of other features to our technical report [fR18] and online documentation[†].**

[†] [github.com/\[AnonymizedforSubmission\]/wiki](https://github.com/[AnonymizedforSubmission]/wiki)

	ID	Dataset	Participated in Design	Position	Years of Experience	Dataset Familiarity
Astro	A1	DES	✓	Researcher	10+	3
	A2	Kepler		Postdoc	8	5
	A3	Kepler		Postdoc	8	5
Genetics	G1	Mouse	✓	GradStudent	4	4
	G2	Cancer		GradStudent	2	2
	G3	Mouse	✓	Professor	10+	2
MatSci	M1	Solvent (8k)	✓	Postdoc	4	5
	M2	Solvent (Full)	✓	Professor	10+	5
	M3	Solvent (Full)	✓	GradStudent	3	5

Table 1: Participant information. The Likert scale used for dataset familiarity ranges from 1 (not familiar) to 5 (extremely familiar).

2.3. Phase II: Evaluation Study

At the end of our participatory design study, we performed a qualitative evaluation to study how analysts interact with different VQS components in practice. In order to make the evaluation more realistic, we invited participants to use datasets that they have a vested interest in exploring to address unanswered research questions. As shown in Table 1, the evaluation study participants included the six scientists from participatory design, along with three additional "blank-slate" participants who had never encountered *zenvisage++* before. **The use of all or a subset of the project participants (e.g., stakeholders) as evaluation participants is common in participatory design [BDI16].** Since the participatory design subjects acted as informants and did not actively try out the system on their own, the evaluation study was the first time that all participants used *zenvisage++* to explore their datasets.

Evaluation study participants were recruited from each of the three aforementioned research groups, as well as domain-specific mailing lists. Prior to the study, we asked potential participants to fill out a pre-study survey to determine eligibility. Eligibility criteria included: being an active researcher in the subject area with more than one year of experience, and having worked on a research project involving data of the same nature used in participatory design. **None of the participants received monetary compensation for the study, as this is not a common practice for participatory design with stakeholders.** As detailed in Table 1, the nine participants brought a total of six different datasets to the study.

At the start, participants were provided with an interactive walk-through explaining **system details** and given approximately ten minutes for a guided exploration of *zenvisage++* with a preloaded real-estate example dataset from Zillow [zil16]. After familiarizing themselves with the tool, we loaded the participant's dataset and encouraged them to talk-aloud during data exploration and use external tools. If the participant was out of ideas, we suggested one of the ten main VQS functionalities that they had not yet used. If any of these operations were not applicable to their specific dataset, they were allowed to skip the operation after having considered how it may or may not be applicable to their workflow. The user study **lasted on average for 63 minutes** and ended after they covered all ten main functionalities. After the study, we asked **participants** open-ended questions about their experience.

3. Participants and Datasets

At the start of our design study, we observed participants as they conducted **contextual inquiry** demonstrating their existing data analysis workflows. Next, we describe our study participants and their preferred analysis workflows.

Astronomy: The Dark Energy Survey is a multi-institution project that surveys 300 million galaxies over 525 nights to study dark energy [Drl17]. The telescope used to survey these galaxies also focuses on smaller patches of the sky on a weekly interval to discover astronomical transients (objects whose brightness changes dramatically as a function of time), such as supernovae or quasars. Their dataset consists of a large collection of **light curves: brightness observations over time, one associated with each astronomical object, plotted as time series**. Over five months, we worked closely with A1, an astronomer on the project's data management team working at a supercomputing facility. Their scientific goal is to identify potential astronomical transients in order to study their properties.

To identify transients, astronomers programmatically generate visualizations of candidate objects with `matplotlib` and visually examine each light curve. While an experienced astronomer who has examined many transient light curves can often distinguish an interesting transient object from noise by sight, manual searching for transients is time-consuming and error prone, since the large majority of the objects are false positives. A1 was interested in VQSs as he recognized how specific pattern queries could help astronomers directly search for these rare transients.

Genetics: Gene expression is a common measurement in genetics obtained via microarray experiments [PS16]. We worked with a graduate student (G1) and professor (G3) at a research university who were using gene expression data to understand how genes are related to phenotypes expressed during early development. Their data consisted of a collection of gene expression profiles over time for mouse stem cells, aggregated over multiple experiments.

Their typical workflow is as follows: G1 first loads the preprocessed gene expression data into a custom desktop application for visualizing and clustering it. After setting several system parameters and executing the clustering algorithm, the overlaid time series for each cluster is displayed on the interface. G1 visually inspects that the patterns in each cluster looks "clean" and checks that the number of outlier genes (i.e., those that do not fall into any of the clusters) is low. If the number of outliers is high or the clustered visualizations look "unclean", she reruns the analysis by increasing the number of clusters. Once the visualized clusters look "good enough", G1 exports the clusters to her downstream regression tasks.

Prior to the study, G1 and G3 spent over a month attempting to determine the best number of clusters based on a series of static visualizations and statistics computed after clustering. While regenerating their results took no more than 15 minutes every time they made a change, the multi-step, segmented workflow meant that all changes had to be done offline. **They were interested in VQSs as interactively querying time series with clustering results could dramatically speed up their collaborative analysis process.**

Material Science: We collaborated with material scientists at a research university who are working to identify solvents for energy

efficient and safe batteries. These scientists work on a large simulation dataset containing chemical properties for more than 280,000 solvents [KKV18]. Each row of their dataset represents a unique solvent with 25 different chemical attributes. We worked closely with a postdoctoral researcher (M1), professor (M2), and graduate student (M3) for over a year to design a sensible way of exploring their data. They wanted to use VQSs to identify solvents that not only have similar properties to known solvents, but are also more favorable (e.g., cheaper or safer to manufacture). To search for these desired solvents, they need to understand how changes in certain chemical attributes affect other properties under specific conditions.

M1 typically starts his data exploration process by iteratively applying filters to a list of potential battery solvents using SQL queries. **Once the remaining solvent list is sufficiently small, each solvent is examined in more detail to factor in its cost and availability to determine experimental feasibility.** They were interested in VQSs as it was impossible for them to uncover hidden relationships between different attributes across large number of solvents manually.

4. System-level Participatory Design Findings

Holzblatt and Jones [HJ93] describes contextual inquiry as a technique that forms the basis for *"developing a system model that will support user's work"* that subsequently *"fosters participatory design"*. Given the need for a VQS highlighted in contextual inquiry interviews, we further distill a list of VQS features by working closely with participants to develop features to address their problems and challenges. In this section, we first reflect on our participatory design (PD) methodology to introduce the PD findings, then we provide a high-level system overview of our PD product, *zenvisage++*.

4.1. The Collaborative Feature Discovery Process

Throughout the PD process, we identified various subtasks based on the scientist's workflow and elicited intermediate feedback. Bodker et. al. [BGK93] cites the importance of encouraging user participation and creativity in cooperative design through different techniques, such as future workshops, critiques, and situational role-playing. Similarly, our PD objective was to collect as many comprehensive list of feature proposals, inclusive across different domains, leading to the added features listed in Table ?? . Given the highly-evolving, unstructured nature of exploratory data analysis (CITE), this technique comes with its advantages and limitations, since users request (or lack thereof) may not always translate to a direct need. For instance, we found that introducing the newly-added features from *zenvisage++* that addressed a particular use case often results in discovering an unexpected practice usage of the feature with other groups of participants. Having feature proposals inspired by multiple use cases can also lead to more generalized design choice. For example, we spoke to astronomers who wanted to eliminate sparse time series from their visual queries. In the same week, we spoke to material scientists who expressed a need for inspecting only solvents with properties above a certain threshold. Through these use cases, data filtering arose as a crucial, common operation that we needed to incorporate into *zenvisage++* in order to support these class of queries.

While our collective brainstorming led to the cross-pollination

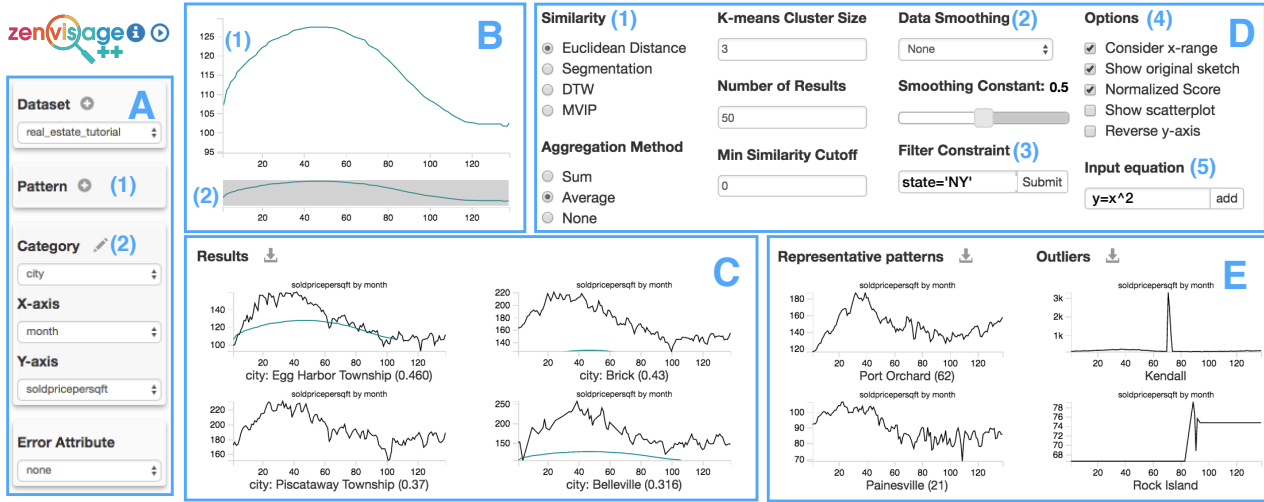


Figure 2: The *zenvisage++* system consist of : (A) data selection panel (where users can select visualized dataset and attributes), (B) query canvas (where the queried data pattern is submitted and displayed), (C) results panel (where the visualizations most similar to the queried pattern is displayed as a ranked list), (D) control panel (where users can adjust various system-level settings), and (E) recommendation (where the typical and outlying trends in the dataset is displayed).

and generalization of features, this technique can also lead to unnecessary features that result in wasted engineering efforts. During the design phase, there were numerous problems and features proposed by participants, but not all were incorporated in the tool. Based on our meeting logs with participants, we found that the reasons for not taking a feature from the design stage to the implementation stage includes:

- **Nice-to-haves:** One of the most common reasons for unincorporated features comes from participant's requests for nice-to-have features. The amount of nice-to-have features that one could envision for the tool is endless. We use two criteria to heuristically judge whether to implement a particular feature:
 1. *Necessity:* Without this feature, can the analyst still work with this dataset using our tool and achieve what they want to do?
 2. *Generality:* If this feature was implemented, will it benefit only this specific use case or be potentially useful for other applications as well?
- **"One-shot" operations:** We decided not to include features that were "one-shot" operations that only needed to be performed once during the analysis workflow. For example, certain data preprocessing operations such as filtering null values only needed to be performed once unlike the data smoothing feature that we added, which was a preprocessing step that could be iteratively performed along with other operations in the VQS. This design choice is specific to our VQS design study.
- **Substantial research or engineering effort:** Some proposed features do not make sense in the context of VQS. For example, the *matsci* group proposed functional fitting to obtain fitting coefficients. Other features requires a completely different set of research questions. For example, the question of how to properly compute similarity between time series with non-uniform number of datapoints arose in the astronomy and genetics use

case, but requires the development of a novel distance metric or algorithm that is out of the scope of the RQs of our design study.

- **Underdeveloped ideas:** Other feature requirements came from casual specification that were underspecified. For example, A1 wanted to look for objects that have deficiency in one band and high emission in another band, but what brightness levels qualifies as a deficiency is ambiguous.

Failure to identify these early signs in the design phase may result in feature implementations that turn out not to be useful for the participants. Given exhaustive nature of Table ??, each motivated by example use cases from one or more domains, we organized the features list in terms of the sensemaking framework described in Section 5 and evaluated its effectiveness in the evaluation study in Section 6.

4.2. System

The *zenvisage++* interface is organized into 5 major regions that dynamically updates upon user interactions. Typically, analysts begin their analysis by selecting the dataset and attribute to visualize in the *data selection panel* (Figure 2A). Then, they can specify a pattern of interest as a query (hereafter referred to as *pattern query*), through either sketching, inputting an equation, uploading a data pattern, or dragging and dropping an existing visualization, displayed on the *query canvas* (Figure 2B). *zenvisage++* performs shape-matching between the queried pattern and other possible visualizations and returns a ranked list of visualizations that are most similar to the queried pattern, displayed in the *results panel* (Figure 2C). At any point during the analysis, analysts can adjust various system-level settings through the *control panel* (Figure 2D) or browse through the list of *recommendations* provided by *zenvisage++* (Figure 2E). Our *zenvisage++* system is open source and available at: [github.com/\[AnonymizedforSubmission\]](https://github.com/[AnonymizedforSubmission]).

Component	Feature	Purpose	Task Example	Similar Features in Past VQSs
Pattern Specification	Sketch-to-query (Figure 2B1)	Freehand sketching for specifying pattern query.	A: Find patterns with a peak and long-tail decay that may be supernovae candidates.	All include sketch canvas except [HS04].
	Input Equation (Figure 2A1)	Specify a exact functional form as a pattern query (e.g., $y=x^2$).	M: Find patterns exhibiting inversely proportional chemical relationship.	—
	Pattern Upload (Figure 2D2)	Upload a pattern consisting of a sequence of points as a query.	A: Find supernovae based on previously discovered sources.	[MVK*11]
Match Specification	Smoothing (Figure 2D2)	Interactively adjusting the level of denoising on visualizations, effectively changing the degree of shape approximation when performing pattern matching.	A, M: Eliminate patterns matched to spurious noise.	Smoothing [MA18] Angular slope queries [HS04] Trend querylines [RLL*05]
	Range Selection (Figure 2B2, D4)	Restrict to query only in specified x/y ranges of interest through brushing selected x-range and filtering selected y-range.	A: Matching only around shape exhibiting a peak. M: Matching only around shape region that exhibit linear or exponential relationships	Text Entry [Wat01, MA18] Min/max boundaries [RLL*05] Range Brushing [HS01]
	Range Invariance (Figure 2D1,4)	Ignoring vertical or horizontal differences in pattern matching through option for x-normalization and y-invariant similarity metrics .	A: Searching for existence of a peak above a certain amplitude. G: Searching for a “generally-rising” pattern.	Temporal invariants [CG16]
View Specification	Data selection (Figure 2A)	Changing the collection of visualizations to iterate over.	M: Explore tradeoffs and relations between physical attributes.	—
	Display control (Figure 2D4)	Changing details of how visualizations should be displayed.	M: Non-time-series data should be displayed as scatterplot.	—
Slice-and-Dice	Filter (Figure 2D3)	Display and query only on data that satisfies the composed filter constraints.	A: Eliminate unlikely candidates by navigating to more probable data regions. M, G: Compare how overall patterns change when filtered to particular data subsets.	—
	Dynamic Class (Figure 9)	Create custom classes of data that satisfies one or more specified range constraints. Display aggregate visualizations of separate classes.	A, M: Examine aggregate patterns for different data subsets.	—
Result Querying	Drag-and-drop (Figure 2C, E)	Querying with any selected result visualization as pattern query (either from recommendations or results).		Drag-and-drop [HS01] Double-Click [CG16]
Recommendation	Representative and Outliers (Figure 2E)	Displaying visualizations of representative trends and outlier instances based on clustering.	A: Examine anomalies and debug data errors through outliers. G, M: Understand representative trends common to this dataset (or filtered subset).	—

Table 2: List of major features incorporated throughout the participatory design study. We organize each feature based on the functional component that it belongs to. Table cells are further colored according to the sensemaking process that each component corresponds to (Blue: Top-down, Yellow: Context creation, Green: Bottom-up). We list the functional purpose of each feature based on how it is implemented in *zenvisage++*, examples use cases from participatory design (**A:** astronomy, **M:** material science, **G:** genetics), and how similar features have been incorporated in past VQSs.

5. The Sensemaking Model for VQSs

To convey how the features in *zenvisage++* addresses the analytical needs posed by each domain, we organize our PD findings into a

sensemaking framework for VQSs. In this section, we first describe the space of problems addressable by VQSs. Then, as shown in Figure 4, we develop a taxonomy for organizing VQS functional-

ties into three sensemaking processes. From top to bottom, we first describe the design objectives of each sensemaking process, then we outline the design challenge addressed by each of the functional components that supports the sensemaking process. The mapping between specific *zenvisage++* features and these functional components and sensemaking processes can be found in Table ??.

5.1. Characterizing the Problem Space for VQSs

Given our earlier description of VQS features organized into components, we now introduce the three sensemaking processes by characterizing how they fit into different problem areas that VQSs are aimed to solve. Visual querying often consists of searching for a desired pattern instance (Z) across a visualization collection specified by some given attributes (X,Y). Correspondingly, we introduce two axes depicting the amount of information known about the visualized attribute and pattern instance.

Along the **pattern instance** axis, the visualization that contains the desired pattern may already be **known** to the analyst, exist as a pattern **in-the-head** of the analyst, or completely **unknown** to the analyst. In the **known** pattern instance region (Figure 3 grey), visualization-at-a-time systems such as Tableau, where analyst manually create and examine each visualization one at a time, is more well-suited than VQSs, since analysts can directly work with the selected instance without the need for visual querying. Inspired by Pirolli and Card's information foraging framework [PC05], which distinguishes between information processing tasks that are *top-down* (from theory to data) and *bottom-up* (from data to theory), we define *top-down pattern specification* as the search-oriented paradigm where analysts query based on their in-the-head pattern (Figure 3 blue) in a fixed collection. On the other hand, in the realm of *bottom-up data-driven inquiry* (Figure 3 green), the pattern of interest is unknown, **external to the user**, and must be driven by recommendations or queries that originate from the data (or equivalently, the visualization). As we will discuss later, this process is crucial but underexplored in past work on VQSs.

The second axis, **visualized attributes**, depicts how much the analyst knows about which X and Y axes they are interested in visualizing. In both the astronomy and genetics use cases, as well as past work in this space, data was in the form of a time series with **known** visualized attributes. In the case of our material science participants, they wanted to explore relationships between different X and Y variables. In this realm of **unknown** attributes, context creation (Figure 3 yellow) is essential for allowing users to pivot across different visualization collections.

5.2. Design Goals for the Sensemaking Processes

After understanding how each sensemaking process fits into the problem space **addressable** by VQSs, we further explore the design objectives and challenges in supporting each sensemaking process, grounded in our collaborative design experience.

Top-down Pattern Specification begins with the user's intuition about how their desired patterns should look like based on 'theory', including visualizations from past experience or abstract conceptions based on external knowledge. The goal of top-down pattern

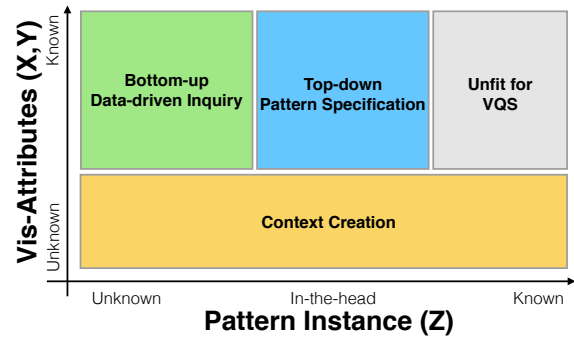


Figure 3: The problem space for VQSs is characterized by how much the analyst knows about the visualized attributes and the pattern instance. Colored areas highlight the three sensemaking processes in VQSs for addressing these characteristic problems. While prior work has focused solely on use cases in the blue region, we envision opportunities for VQSs beyond this to a larger space of use cases covered by the yellow and green regions.

specification is to address the *which* question of visual sensemaking: *which pattern instance exhibits this pattern?* Based on this preconceived notion of what to search for, the design challenge is to translate the query in the analyst's head to a query executable by the VQS. In the Figure 4 taxonomy, this includes both components for specifying the pattern, as well as controls governing the underlying algorithm of how shape-matching is performed. For example, A1 knows intuitively what a supernovae pattern looks like and the detailed constraints on the shape, such as the width and height of the peak or the level of error tolerance for defining a match. He can search for transient patterns through sketching, select the option to ignore differences on the x axis, and changes the similarity metric for flexible matching.

Bottom-up data-driven inquiry is a browsing-oriented sensemaking process that goes from data to theory to addresses the *what* questions in the sensemaking process. For example, genetics participants do not have a preconceived knowledge of what to search for in the dataset. They were mostly interested in *what types of patterns exist in the dataset* through representative trends, as a means to jumpstart further queries. The design challenge include developing the right set of 'stimuli' that could provoke further data-driven inquiries, as well as low-effort mechanisms to search via these results.

Context Creation addresses the *where* question of sensemaking by enabling analysts to navigate across different parts of the visualization collection to learn about *where in the dataset do the patterns of interest lie*. For example, material scientists often do not start with a pattern in-the-head, but recognize salient trends such as inverse correlation or linear correlation. They switch between different visualized attributes or dynamic classes to study their data from alternative perspectives. The design challenge of context creation is to develop features that act as a 'lens': navigating users to desired data subsets, visualizing and comparing how the data changes between the different lenses, and ensuring that context is dynamically reflected across other VQS functionalities.

The three aforementioned sensemaking processes are akin to the well-studied sensemaking paradigms of search, browse, and faceted

navigation on the Web [Hea09, OC03]. Due to each of their advantages and limitations given different information seeking tasks, search interfaces have been designed to support all three complementary acts and transition smoothly between them to combine the strength of all three paradigms. Similarly for VQSs, our main design objective in developing *zenvisage++* is to integrate all the three sensemaking in the same system. As we discover in the evaluation study in the following section, this integration encourages and accelerates the process of visualization discovery.

5.3. Functional Components of VQSs

Here, we discuss the motivation for each functional component in the lower-level of our Figure 4 taxonomy and how they address specific challenges posed by the problem and dataset characteristics from each domain.

Pattern Specification interfaces allow users to submit exact descriptions of a pattern query with the VQS returning a list of most similar matches. This is useful when the dataset contains *large numbers of potentially-relevant pattern instance*. Since it is often difficult to sketch precisely, additional characteristics of the pattern query can be used to further winnow our undesired matches (e.g., pattern query expressible in a functional form, or has specific shape characteristics).

Match Specification addresses the well-known problem in VQSs where pattern queries are imprecise [CG16, HF09, EZ15] by allowing users to clarify how matching should be performed. Match specification is useful when the dataset is *noisy* (i.e., containing large numbers of false-positives that could be matched). When the pattern query additional constraints, adjusting match specification to prune away these false-positives help reveal true candidates.

View specification settings alter the specifications for all of the candidate visualizations being explored on the VQS. The ability to work with different collections of visualizations is useful when the dataset is *multidimensional* and the axes of interest is unknown. Modifying the view specification offers analysts different perspectives on the data to locate visualization collections of interest.

Slice-and-Dice empowers users to navigate and compare different collections of visualizations constructed from different portions of the data. Slice-and-dice is useful when the dataset has *large numbers of non-visualized attributes* that may be related to the visualized attributes (e.g., geographical location may influence the time series pattern for housing prices). Analysts can either make use of pre-existing knowledge regarding these ‘support’ attributes to navigate to a data region that is more likely to contain the desired pattern (e.g., filtering to popular cities such as New York to find expensive houses) or discover unknown patterns and relationships between different data subsets (e.g., housing prices is lower around winter than compared to summer).


Result querying allows users to submit a query based on the results, essentially asking for patterns similar to the selected data pattern. Typically, analyst associate selected visualizations with some *semantic or visual properties* and make use of results querying to understand characteristic properties of similar instances.

Recommendation displays visualizations that may be of interest to users based on the data context. Representative trends and outliers are useful when a *small number of common patterns* is exhibited in the dataset. Understanding *characteristic* patterns in dataset can help analysts discover other pattern queries of interest to jumpstart further queries.

6. Evaluation Study Findings

Based on audio, video screen capture, and click-stream logs recorded during our evaluation study, we performed thematic analysis via open coding and categorized every event with a coded label. Event codes included specific feature usage, insights, provoked actions, confusion, request for functionalities unaddressed by the system, and use of external tools, detailed in Appendix B. To characterize the usefulness of each feature, we further labeled whether each feature was useful to a particular participant’s analysis. A feature was deemed *useful* if the feature was either used in a sensible and meaningful way during the study, or has envisioned usage outside of the constrained time limit during the study (e.g., if data was available or downstream analysis was conducted). We derived these labels from the study transcript to circumvent self-reporting bias, which can often artificially inflate the usefulness of the feature under examination. In this section, we will apply our thematic analysis results to understand how each sensemaking process occurs in practice.

6.1. The Ineffectiveness of Sketch

To our surprise, despite the prevalence of sketch-to-query systems in the literature, only two out of our nine participants found it useful to directly sketch *desired* pattern onto the canvas. The *reason why most* participants did not find *sketching useful* was that they often do not start their analysis with a specific pattern in mind. Instead, their intuition about what to query is derived from other visualizations they encounter during exploration, in which case it makes more sense to query using those visualizations as examples directly (e.g., by dragging and dropping that visualization onto the *canvas to submit the query*). Even if a user has a pattern in mind, translating that pattern into a sketch is often hard to do. For example, A2 wanted to search for a highly-varying signal enveloped by a sinusoidal pattern indicating planetary rotation , which is hard to draw by hand.

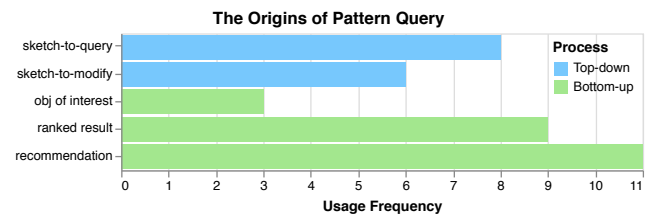


Figure 5: The number of times a pattern query originates from one of the workflows. We find that pattern queries are *far* more commonly generated via bottom-up than top-down processes.

Given these initial findings, we further investigated where the pattern on the canvas typically originates, as presented in Figure 5.

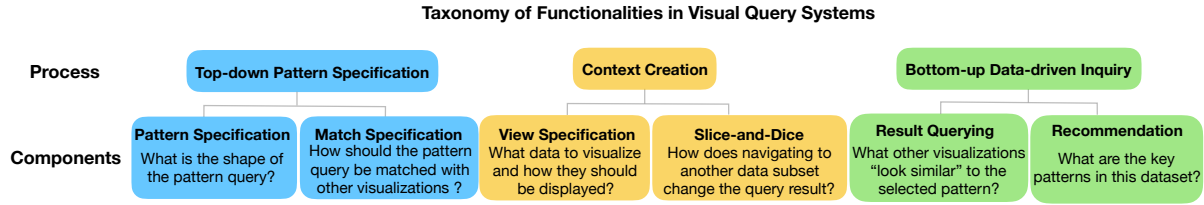


Figure 4: Taxonomy of functionalities in VQSs. Each of the three sensemaking process is broken down into key functional components in VQSs. We list the types of questions addressed by each component from a system's perspective.

Pattern queries can be generated by either top-down (sketching) or bottom-up (drag-and-drop) processes, driven by various different querying intentions. Within top-down processes, a pattern query could arise from users directly sketching a new pattern (sketch-to-query) or by modifying an existing sketch (sketch-to-modify). For example, M2 first sketched a pattern to find solvent classes with anticorrelated properties without much success in returning a desired match. So he instead dragged and dropped one of the peripheral visualizations similar to his desired visualization and then smoothed out the noise in the visualization yielding a straight line, as shown in Figure 6 (left). M2 repeated this workflow twice in separate occurrences during the study and was able to derive insights from the results. Likewise, Figure 6 (right) illustrates how A3 first picked out a regular pattern (suspected star spot), then modified it slightly so that the pattern looks more irregular (to find pulsating stars). As described in the following section, bottom-up pattern queries can come from either the ranked list of results, recommendations, or by selecting a particular object of interest as a drag-and-drop query. Figure 5 shows that bottom-up processes are more common than top-down processes for generating a pattern query.

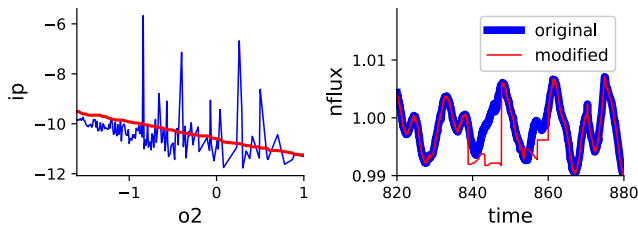


Figure 6: Canvas traces from M2 (left) and A3 (right) during the study demonstrating query modification. The original drag-and-dropped query is shown in blue and sketch-modified queries in red.

The lack of practical use of top-down pattern specification is also reflected in the fact that none of the participants queried using an equation. In both astronomy and genetics, the visualization patterns resulting from complex physical processes that could not be written down as an equation analytically. Even in the case of material science when analytical relationships do exist, it is challenging to formulate functional forms in a prescriptive manner.

Our findings suggest that while sketching is an useful construct for people to express their queries, the existing ad-hoc, sketch-only model for VQSs is insufficient without data examples that can help analysts jumpstart their exploration. In fact, from Figure 5, we can

see that sketch-to-query was only used 8 times, while the remaining querying modalities were used 29 times altogether, more than three times as much as sketch-to-query. This finding has profound implications on the design of future VQSs, since Table ?? suggests that past work have primarily focused on optimizing top-down process components, without considering how useful these features are in real-world analytic tasks. We suspect that these limitations may be why existing VQSs are not commonly adopted in practice.

6.2. Context Creation and Bottom-up Applications

As alluded to earlier, bottom-up data-driven inquiries and context creation are far more commonly used than top-down pattern specification when users have no desired patterns in mind, which is typically the case for exploratory data analysis. In particular, we find that top-down approaches were only useful for 29% of the use cases, whereas it was useful for 70% of the use cases for bottom-up approaches and 67% for context creation[‡]. We now highlight some of the exemplary workflows demonstrating the efficacy of the latter two sensemaking processes.

As shown in Figure 5, the most common use of bottom-up querying is via recommended visualizations. For example, G2 and G3 identified that the three representative patterns recommended in *zenvisage++* corresponded to the same three groups of genes discussed in a recent publication [GSC*17]: induced genes (profiles with expression levels going up), repressed genes (starting high then decreasing), and transients (rising first then dropping at another time point). The clusters provoked G2 to generate a hypothesis regarding the properties of transients: "Is that because all the transient groups get clustered together, or can I get sharp patterns that rise and ebb at different time points?" To verify this hypothesis, G2 increased the parameter controlling the number of clusters and noticed that the clusters no longer exhibited the clean, intuitive patterns he had seen earlier. G3 expressed a similar sentiment and proceeded by inspecting the visualizations in the cluster via drag-and-drop. He found a group of genes that all transitioned at the same timestep, while others transitioned at different timesteps.

By browsing through the ranked list of results in *zenvisage++*, participants were also able to gain a peripheral overview of the data and spot anomalies during exploration. For example, A1 spotted time series that were too faint to look like stars after applying the

[‡] See Appendix B for details on how this measure is computed.

filter CLASS_STAR=1, which led him to discover that all stars have been mislabeled with CLASS_STAR=0 as 1 during data cleaning.

Past studies in visual analytics have shown that it is important to design features that enable users to select relevant subsets of data [AES05, HS12]. Context creation in VQSs enables users to change the ‘lens’ by which they look through the data when performing visual querying, thereby creating more opportunities to explore the data from different perspectives. All participants found at least one of the features in context creation to be useful.

Both A1 and A2 expressed that interactive filtering enabled them to test conditions and tune values that they would not have otherwise modified, effectively lowering the barrier between the iterative hypothesize-then-compare cycle during sensemaking. During the study, participants used filtering to address questions such as: *Are there more genes similar to a known activator when we subselect only the differentially expressed genes?* (G2) or *Can I find more supernovae candidates if I query only on objects that are bright and classified as a star?* (A1). Three participants had also used filtering as a way to pick out individual objects of interest to query with, as shown in Figure 5. For example, G2 set the filter as gene=9687 and explained that since “*this gene is regulated by the estrogen receptor, when we search for other genes that resemble this gene, we can find other genes that are potentially affected by the same factors.*”

While filtering enabled users to narrow down to a selected data subset, dynamic class creation enabled users to compare relationships between multiple attributes and subgroups of data. For example, M2 divided solvents in the database into eight different categories based on voltage properties, state of matter, and viscosity levels, by dynamically setting the cutoff values on the quantitative variables to create these classes. By exploring these custom classes, M2 discovered that the relationship between viscosity and lithium solvation energy is independent of whether a solvent belongs to the class of high voltage or low voltage solvents. He cited that dynamic class creation was central to learning about this previously-unknown attribute properties:

All this is really possible because of dynamic class creation, so this allows you to bucket your intuition and put that together. [...] I can now bucket things as high voltage stable, liquid stable, viscous, or not viscous and start doing this classification quickly and start to explore trends. [...] look how quickly we can do it!

6.3. Sensemaking in VQS Workflows

Given our observations so far as to how participants make use of each sensemaking process in practice, we further investigate the interplay between these sensemaking processes in the context of an analysis workflow. The event sequences from the evaluation study consist of labels describing when specific features were used. Using the taxonomy in Section 4, we map each usage of a feature to one of the three sensemaking processes. Each participant’s event sequence is divided into sessions, each indicating a separate lines of inquiry during the analysis. Based on these event sequences—one for each session, we compute the aggregate state transition probabilities (shown as edge weights in Figure 7) to characterize how participants from each domain move between different sensemaking processes. For example, in material science, bottom-up exploration leads to context creation 60% of the time and to top-down pattern-specification the rest of the time. Self-directed edges indicate the

probability that the participant would continue with the same type of sensemaking process. For example, when an astronomer performs top-down pattern specification, it is followed by another top-down specification 64% of the time and context creation the rest of the time, but never followed by a bottom-up processes. This high self-directed transition probability reflects how astronomers often need to iteratively refine their top-down query through pattern or match specification when looking for a specific pattern.

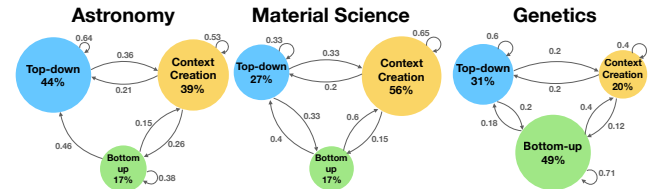


Figure 7: Markov models computed based on the evaluation study event sequences, with edges denoting the probability that a participant in the particular domain will go from one sensemaking process to the next. Nodes are scaled according to the eigenvector centrality, which represents the percentage of time users would spend in a particular sensemaking process in steady state.

To study how important each sensemaking process is for participant’s overall analysis, we compute the eigenvector centrality of each graph, displayed as node labels in Figure 7. These values represent the percentage of time the participants spend in each of the sensemaking processes when the transition model has evolved to a steady state [Pie11]. Given that nodes in Figure 7 are scaled by this value, in all domains, we observe that there is always a prominent node connected to two less prominent ones—but it is also clear that all three nodes are essential to all domains. Our observation demonstrates how participants often construct a central workflow around a main sensemaking process and interleave variations with the two other processes as they iterate on the analytic task. For example, material scientists focus on context creation 56% of the time, mainly through dynamic class creation, followed by bottom-up inquiries (such as drag-and-drop) and top-down pattern specification (such as sketch modification). The central process adopted by each domain is tightly coupled with characteristics of the analytic challenges associated with their subject area, as described earlier. For example, without an initial query in-the-head, geneticists relied heavily on bottom-up querying through recommendations to jumpstart their queries.

The Markov transition model exemplifies how participants adopted a diverse set of workflows based on the unique set of research questions they brought to the study. The bi-directional and cyclical nature of the transition graphs in Figure 7 highlight how the three sensemaking processes do not simply follow a linear progression, going from unknown to known in the Figure 3 problem space. Instead, the high connectivity of the transition model illustrates how these three equally-important processes form a sensemaking loop, representing iterative acts of dynamic foraging and hypothesis generation. This flexibility is enabled by the diverse set of potential workflows that could be constructed in an integrative VQS like *zenvisage++*, for addressing a wide range of analytical inquiries.

6.4. Limitations

Although evidence from our evaluation study suggests that direct sketch is inefficient, we have not performed controlled studies with a sketch-only system as a baseline to validate this hypothesis. The goal of our study is to uncover qualitative insights that might reveal why VQSs are not widely used in practice; further validation of specific findings is out of the scope of this paper. While we have generalized our findings by employing three different and diverse domains (see Figure 7), our case studies have so far been focused on scientific data analysis, as a first step towards greater adoption of VQSs. Other potential domains that could benefit from VQSs include: financial data for business intelligence, electronic medical records for healthcare, and personal data for “Quantified Self”. These different domains may each pose different sets of challenges unaddressed by the findings in this paper, pointing to a promising direction for future work.

7. Conclusion

While VQSs hold tremendous promise in accelerating data exploration, they are rarely used in practice. In this paper, we worked closely with analysts from three diverse domains to characterize how VQSs can address their analytic challenges, collaboratively design VQS features, and evaluate how VQS functionalities are used in practice. Participants were able to use our final deployed system, *zenvisage++*, for discovering desired patterns and trends, and obtaining valuable insights to address unanswered research questions. Grounded in these experiences, we developed a sensemaking model for how analysts make use of VQSs. Contrary to past work, we found that sketch-to-query is not as effective in practice as past work may suggest. Beyond sketching, we find that each sensemaking process fulfills a central role in participants’ analysis workflows to address their high-level research objectives. We advocate that future VQSs should invest in understanding and supporting all three sensemaking processes to effectively ‘close the loop’ in how analysts interact and perform sensemaking with VQSs. While more work certainly remains to be done, by contributing to a better understanding of how VQSs are used in practice across domains, our paper can also serve as a roadmap for broader adoption of VQSs, and hopefully trigger exploration of novel use cases for these tools.

Appendix A: Artifacts from Participatory Design

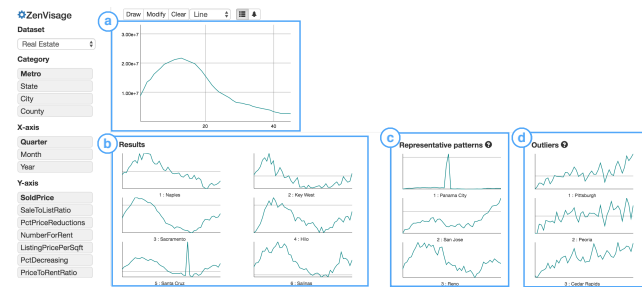


Figure 8: The existing *zenvisage* prototype allowed users to sketch a pattern in (a), which would then return (b) results that had the closest Euclidean distance from the sketched pattern. The system also displays (c) representative patterns obtained through K-Means clustering and (d) outlier patterns to help the users gain an overview of the dataset.

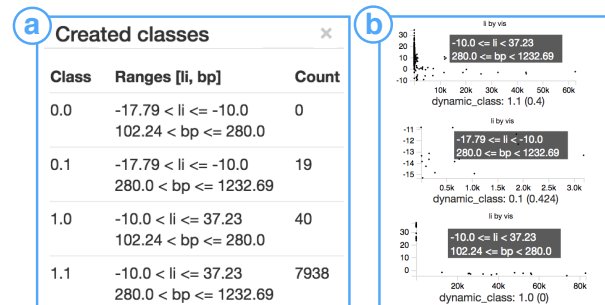


Figure 9: Example of dynamic classes. (a) Four different classes with different Lithium solvation energies (li) and boiling point (bp) attributes based on user-defined data ranges. (b) Users can hover over the visualizations for each dynamic class to see the corresponding attribute ranges for each class. The visualizations of dynamic classes are aggregate across all the visualizations that lie in that class based on the user-selected aggregation method.



Figure 10: Examples of the scientists’ original workflow: a) A1 examines a light curve manually using the Jupyter notebook environment, b) G2 uses a domain-specific software to examine clustering outputs.

Appendix B: Evaluation Study Analysis Details

We analyzed the transcriptions of the evaluation study recordings through open-coding and categorized every event in the user study using the following coding labels:

- Insight (Science) **[IS]**: Insight that connected back to the science (e.g. “This cluster resembles a repressed gene.”)
- Insight (Data) **[ID]**: Data-related insights (e.g. “A bug in my data cleaning code generated this peak artifact.”)
- Provoke (Science) **[PS]**: Interactions or observations that provoked a scientific hypothesis to be generated.
- Provoke (Data) **[PD]**: Interactions or observations that provoked further data actions to continue the investigation.
- Confusion **[C]**: Participants were confused during this part of the analysis.
- Want **[W]**: Additional features that participant wants, which is not currently available on the system.
- External Tool **[E]**: The use of external tools outside of *zenvisage++* to complement the analysis process.
- Feature Usage **[F]**: One of the features in *zenvisage++* was used.
- Session Break **[BR]**: Transition to a new line of inquiry.

Domain	IS	ID	PS	PD	C	W	E	BR	F
astro	4	12	13	57	2	18	20	22	67
genetics	8	12	7	35	4	13	1	21	52
mat sci	14	8	7	44	8	11	3	12	48

Table 3: Count summary of thematic event code across all participants of the same subject area.

In addition, based on the usage of each feature during the user study, we categorized the features into one of the three usage types:

- Practical **[P]**: Features used in a sensible and meaningful way.
- Envisioned usage **[E]**: Features which could be used practically if the envisioned data was available or if they conducted downstream analysis, but was not performed due to the limited time during the user study.
- Not useful **[N]**: Features that are not useful or do not make sense for the participant’s research question and dataset.

The feature usage labels for each user is summarized in Figure 11. A feature is regarded as *useful* if it has a **P** or **E** code label. Using the matrix from Figure 11, we compute the percentage of useful features for each sensemaking process as:

$$\frac{\text{\# of useful features in process}}{\text{total \# of features in process} \times \text{total \# of users}}$$

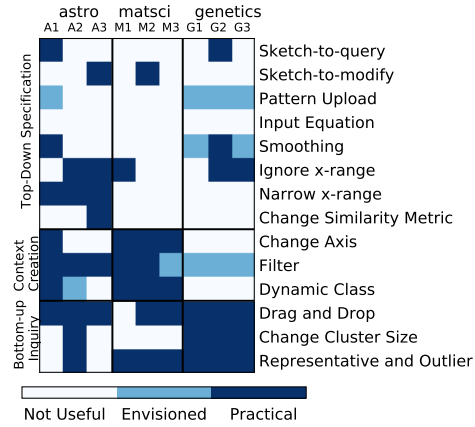


Figure 11: Heatmap of features categorized as practical usage (P), envisioned usage (E), and not useful (N). The columns are arranged in the order of subject areas and the features are arranged in the order of the three foraging acts. We find that participants preferred to query using bottom-up methods such as drag-and-drop over top-down approaches such as sketching or input equations. Participants found that context creation via filter constraints and dynamic class creation were powerful ways to compare between subgroups or filtered subsets.

References

- [AES05] AMAR R., EAGAN J., STASKO J.: Low-level components of analytic activity in information visualization. In *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on* (2005), IEEE, pp. 111–117. 9
- [BDI16] BOSSEN C., DINDLER C., IVERSEN O. S.: Evaluation in participatory design: A literature survey. In *Proceedings of the 14th Participatory Design Conference: Full Papers - Volume 1* (New York, NY, USA, 2016), PDC '16, ACM, pp. 151–160. 3
- [BGK93] BODKER S., GRONBAEK K., KYNG M.: Cooperative design: Techniques and experiences from the scandinavian scene. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 1993, ch. 8. 4
- [CAM*16] CIOLFI L., AVRAM G., MAYE L., DULAKE N., MARSHALL M. T., VAN DIJK D., MCDERMOTT F.: Articulating Co-Design in Museums: Reflections on Two Participatory Processes. *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing - CSCW '16* (2016), 13–25. 3
- [CG16] CORRELL M., GLEICHER M.: The semantics of sketch: Flexibility in visual query systems for time series data. In *Visual Analytics Science and Technology (VAST), 2016 IEEE Conference on* (2016), IEEE, pp. 131–140. 1, 2, 7, 13
- [Dr17] DRILICA WAGNER ET AL.: Dark Energy Survey Year 1 Results: Photometric Data Set for Cosmology. 4
- [EZ15] EICHMANN P., ZGRAGGEN E.: Evaluating Subjective Accuracy in Time Series Pattern-Matching Using Human-Annotated Rankings. *Proceedings of the 20th International Conference on Intelligent User Interfaces - IUI '15* (2015), 28–37. 1, 2, 7
- [fR18] FOR REVIEW A.: You can't always sketch what you want: Understanding sensemaking in visual query systems (technical report). 3
- [GSC*17] GLOSS B. S., SIGNAL B., CHEETHAM S. W., GRUHL F., KACZOROWSKI D. C., PERKINS A. C., DINGER M. E.: High resolution temporal transcriptomics of mouse embryoid body development reveals complex expression dynamics of coding and noncoding loci. *Scientific Reports* 7, 1 (2017), 6731. 8
- [Hea09] HEARST M. A.: *Search User Interfaces*, 1st ed. Cambridge University Press, New York, NY, USA, 2009. 7

- [HF09] HOLZ C., FEINER S.: Relaxed selection techniques for querying time-series graphs. In *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2009), UIST '09, ACM, pp. 213–222. 1, 2, 7
- [HJ93] HOLTZBLATT K., JONES S.: Contextual inquiry: A participatory technique for system design. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 1993, ch. 9. 4
- [HS01] HOCHHEISER H., SHNEIDERMAN B.: Interactive exploration of time series data. In *Discovery Science* (Berlin, Heidelberg, 2001), Springer, pp. 441–446. 2, 13
- [HS04] HOCHHEISER H., SHNEIDERMAN B.: Dynamic query tools for time series data sets: Timebox widgets for interactive exploration. *Information Visualization* 3, 1 (2004), 1–18. 1, 2, 13
- [HS12] HEER J., SHNEIDERMAN B.: A taxonomy of tools that support the fluent and flexible use of visualizations. *Interactive Dynamics for Visual Analysis 10* (2012), 1–26. 9
- [IZCC08] ISENBERG P., ZUK T., COLLINS C., CARPENDALE S.: Grounded Evaluation of Information Visualization. *Proceedings of the 2008 conference on BEyond time and errors novel evaluation methods for Information Visualization - BELIV '08* (2008), 1. 2
- [KKV18] KHETAN A., KRISHNAMURTHY D., VISWANATHAN V.: Towards synergistic electrode-electrolyte design principles for nonaqueous li-o2 batteries. 4
- [LBI*12] LAM H., BERTINI E., ISENBERG P., PLAISANT C., CARPENDALE S.: Empirical studies in information visualization: Seven scenarios. *IEEE Transactions on Visualization and Computer Graphics* 18, 9 (2012), 1520–1536. 2
- [MA18] MANNINO M., ABOUZIED A.: Expressive Time Series Querying with Hand-Drawn Scale-Free Sketches. 1–12. 1, 2, 13
- [MS12] MICHAEL SEDLMIR MIRIAH MEYER T. M.: Design study methodology: Reflections from the trenches and the stacks. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (Dec 2012), 2431–2440. 2
- [Mun09] MUNZNER T.: A nested model for visualization design and validation. *IEEE transactions on visualization and computer graphics* 15, 6 (2009). 2
- [MVK*11] MOHEBBI M., VANDERKAM D., KODYSH J., SCHONBERGER R., CHOI H., KUMAR S.: Google correlate whitepaper. 1, 2, 13
- [OC03] OLSTON C., CHI E. H.: ScentTrails: Integrating Browsing and Searching on the Web. *ACM Transactions on Computer-Human Interaction* 10, 3 (2003), 177–197. 7
- [PC05] PIROLLO P., CARD S.: The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of International Conference on Intelligence Analysis* (2005), vol. 5, pp. 2–4. 1, 6
- [Pie11] PIERRE B.: *Markov chains: Gibbs fields, Monte Carlo simulation, and queues*. Springer, 2011. 9
- [Pla04] PLAISANT C.: The challenge of information visualization evaluation. In *Proceedings of the Working Conference on Advanced Visual Interfaces* (2004), ACM, pp. 109–116. 2
- [PS16] PENG P. C., SINHA S.: Quantitative modeling of gene expression using dna shape features of binding sites. *Nucleic Acids Research* 44, 13 (2016), e120. 4
- [RLL*05] RYALL K., LESH N., LANNING T., LEIGH D., MIYASHITA H., MAKINO S.: Querylines: approximate query for visual browsing. In *CHI'05 Extended Abstracts on Human Factors in Computing Systems* (2005), ACM, pp. 1765–1768. 1, 2, 13
- [SG91] SUSANNE BODKER, GRØNBÆK K.: Cooperative Prototyping -. *International Journal of man-machine studies* (1991), 1–23. 3
- [SKL*16] SIDDIQUI T., KIM A., LEE J., KARAHALIOS K., PARAMESWARAN A.: Effortless data exploration with zenvisage: an expressive and interactive visual analytics system. *Proceedings of the VLDB Endowment* 10, 4 (2016), 457–468. 1, 3
- [SLK*17] SIDDIQUI T., LEE J., KIM A., XUE E., YU X., ZOU S., GUO L., LIU C., WANG C., KARAHALIOS K., PARAMESWARAN A.: Fast-forwarding to desired visualizations with zenvisage. In *The biennial Conference on Innovative Data Systems Research (CIDR)* (2017). 1, 3
- [SP06] SHNEIDERMAN B., PLAISANT C.: Strategies for evaluating information visualization tools: multi-dimensional in-depth long-term case studies. In *Proceedings of the 2006 AVI workshop on BEyond time and errors: novel evaluation methods for information visualization* (2006), ACM, pp. 1–7. 2
- [Wat01] WATTENBERG M.: Sketching a graph to query a time-series database. In *CHI'01 Extended Abstracts on Human factors in Computing Systems* (2001), ACM, pp. 381–382. 1, 2, 13
- [zil16] Zillow. www.zillow.com, 2016. Accessed: February 1, 2016. 3