

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

---

## 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 not able 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 [?, ?, ?, ?, ?, ?, ?, ?]. 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 [?, ?, ?, ?, ?].

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, cognitive walkthroughs, 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* [?, ?], 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 [?]. 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

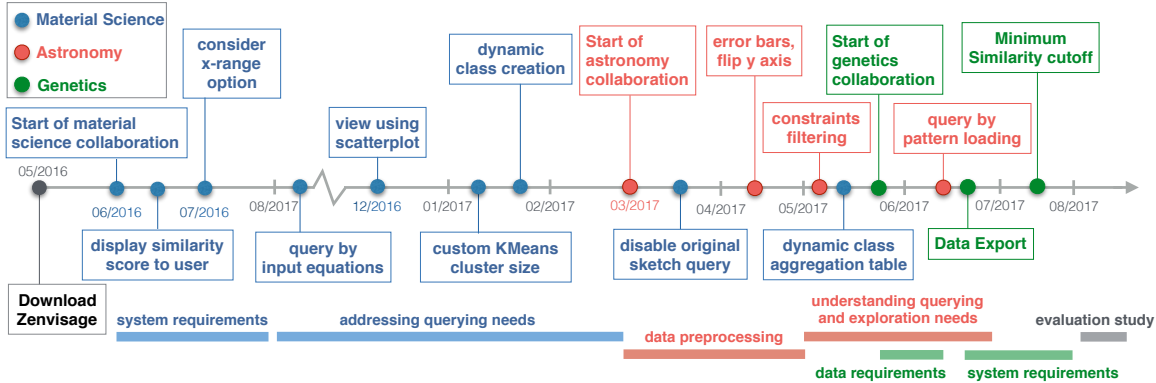


Figure 1: Timeline for progress in participatory design studies.

ignoring the other two processes that are crucial for the needs in all three domains.

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,

- a full-fledged VQS, *zenvisage++*, capable of facilitating rapid hypothesis generation and insight discovery,
- evaluation 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 [?, ?], where the query specification mechanism is a rectangular box, filtering out all of the time series that does not pass through it, QuerySketch [?] and Google Correlate [?], 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 [?, ?, ?] and improving the expressiveness of sketched queries through finer-grained specification interfaces and pattern-matching algorithms [?, ?].

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 [?, ?], 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 [?], 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 [?, ?, ?] 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

VQs as part of the *data/operation abstraction design*. Finally, we validated the abstraction design with grounded evaluation [?, ?], 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 VQs as well as diversity in use cases. Six scientists from three research groups participated in the design of *zenvisage++*. On average, participants had more than 6 years of research experience working in their respective fields. Via interviews and cognitive walkthroughs **using participant's original analysis workflow**, we identified the needs and challenges of these use cases.

For the participatory design study, we built on an existing VQS, *zenvisage* [?, ?], 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 2). **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 [?]. Our motivation for providing a functional prototype at the beginning of the participatory design sessions is to showcase capabilities of VQs. Especially since VQs 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 features** could help address their research questions. **Since essential features that were crucial for effective exploration were lacking in the original *zenvisage* and under development in *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 the intermediate versions of *zenvisage++* were demonstrated to the participants to elicit feedback on how these VQS features could better support their scientific goals.** 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 the new version of our VQS, *zenvisage++*, described more in Section 4.2.

## 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 [?]. 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, as this is not a common practice for participatory design with stakeholders.** [?, ?] the nine participants brought a total of six different datasets to the study. The research questions and objectives of the participants were diverse even among the same subject area. Examples included understanding gene expression profiles of breast cancer cells after a particular treatment and comparing common patterns among stars that exhibit planetary transits versus stars that do not.

At the start, participants were provided with an interactive walkthrough explaining the system details and given approximately ten minutes for a guided exploration of *zenvisage++* with a preloaded real-estate example dataset from Zillow [?]. 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 ended after they covered all ten main functionalities. On average, data exploration lasted for 63 minutes. After the study, we asked them open-ended questions about their experience.

## 3. Participants and Datasets

At the start of our design study, we observed participants as they conducted cognitive walkthroughs 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 [?]. 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 brightness observations over time, one associated with each astronomical object, called a *light*

	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).

curve, and plotted as a time series. For 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 [?]. 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. The team were interested in VQSs as they saw how interactively querying time series with clustering results could dramatically speed up their collaborative analysis process.

**Material Science:** We collaborated with material scientists at a re-

search 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 [?]. 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. When the remaining list of the solvents is sufficiently small, he examines each solvent in more detail to factor in the cost and availability to determine experimental feasibility. The scientists were interested in VQSs as it was impossible for them to uncover hidden relationships between different attributes across large number of solvents manually.

#### 4. Participatory Design Findings

All of the three domains described in the previous section recognized the need for a VQS. As discussed in Section 2, we worked closely with participants to develop features to address their problems and challenges. In this section, we first provide a high-level system overview of *zenvisage++*. Through our participatory design findings, we develop a taxonomy for organizing these functionalities into three sensemaking processes, as shown in Figure 3. Starting from the bottom level of the taxonomy, we first materialize what each component in our taxonomy entails, then we proceed onto the upper level of the taxonomy, describing the space of problems addressable by each sensemaking process and their design objectives.

##### 4.1. 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 query of interest, 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). Given the space limitation of our paper, we have focussed our discussion major *zenvisage++* functionalities relevant to the study findings, and defer the details of other *zenvisage++* features to our technical report [?] and documentation in our open-source online repository<sup>†</sup>.

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



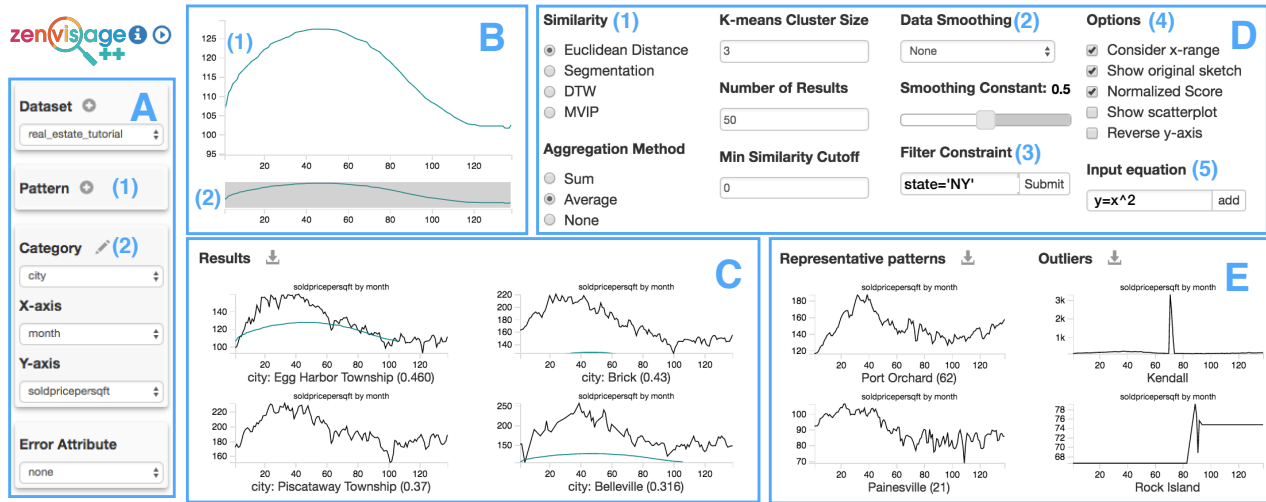


Figure 2: The *zenvisage++* system, including: the ability to query via (a) a sketch, (b) input equations, (i) uploaded patterns, or (j) drag and drop; (c) data smoothing; query specification mechanisms including (d) x-range selection and filtering, (e) x-range invariance, (f) similarity metric selection, (g) filtering, and (h) dynamic class creation; recommendation of (k) representative and (l) outlier trends.

#### Taxonomy of Functionalities in Visual Query Systems

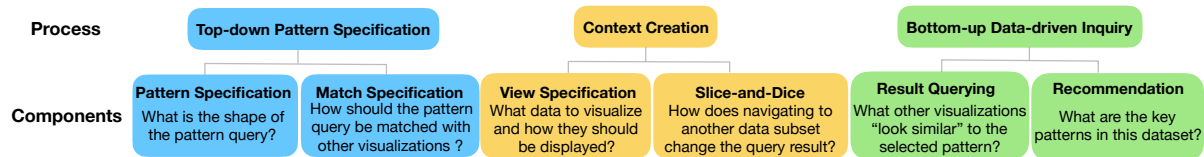


Figure 3: Taxonomy of functionalities in VQSs. Each of the three sensemaking process is broken down into key components in VQSs.

## 4.2. Components Emerging from Participatory Design

Here, we discuss the purpose of the components in the lower-level of our Figure 3 taxonomy, motivating use cases collected over the course of participatory design, and how these themes instantiate into corresponding features in *zenvisage++* and other existing VQSs. Each of the features described (labelled F\*) correspond directly to a use case challenge (labelled C\*) highlighted in the list above it.

### I) Pattern Specification

**Purpose:** Specify exact descriptions of a pattern as a query (hereafter referred to as *pattern query*), with the VQS returning a list of most similar matches.

#### Motivating Use Case Challenges:

- C1:** General need for an intuitive mechanism for specifying desired patterns.
- C2:** Material scientists were interested in finding solvents with known analytical models describing chemical relationships.
- C3:** Astronomers were interested in specifying the exact width of a supernovae light curve, which is characteristic to the radioactive decay rate of its chemical signature [?], to distinguish patterns of interest from noise.

#### Instantiated Feature:

- F1:** Almost all existing VQSs, including *zenvisage++*, support freehand sketching for specifying a desired query pattern on a virtual canvas (Figure 2B1).
- F2:** The above use cases highlights the common difficulty of

sketching precisely. *zenvisage++* allows users to specify an exact functional form (e.g.,  $y=x^2$ ) as a pattern query (Figure 2D5).

- F3:** *zenvisage++* also enables users to upload a pattern consisting of a sequence of points as a query (Figure 2A1). The pattern upload functionality is also available in Google Correlate [?].

### II) Match Specification

**Purpose:** Past work has shown that pattern queries can be imprecise [?, ?, ?]. To this end, VQSs need to support mechanisms for clarifying the interpretation of the sketch (i.e., how matching should be performed).

#### Motivating Use Case Challenges:

- C1:** Dense and noisy observational data in astronomy and material science makes it difficult to match with other patterns exactly.
- C2:** When analyzing line charts, there are often specific ranges with domain significance that users want to restrict their search within or outside of these ranges (e.g., a known period of time when an imaging equipment is malfunctioning).
- C3:** Astronomers and geneticists were interested in qualitative patterns, such as the existence of a peak above a certain amplitude or a 'generally rising' profile, without regards to the exact time when the event occurs. The default Euclidean metric unnecessarily penalizes unaligned time-series of interest.

#### Instantiated Feature:

**F1:** *zenvisage++* supports an interface for users to adjust smoothing algorithms and parameters on-the-fly to update the resulting visualizations accordingly (Figure 2D2). In addition to denoising the data, data smoothing effectively allows users to change the degree of shape approximation they would like to apply to all visualizations when performing pattern matching. Smoothing is also supported in Qetch [?]. Other interfaces have also developed constrained sketching mechanisms to allow users to partially specify certain shape characteristics, such as angular slope queries [?] or piecewise trend query-lines [?]. Smoothing was chosen over these other interfaces for approximating key patterns in the data, since it was a familiar preprocessing step in our study participants' workflow.

**F2:** To restricted search to selected ranges in *zenvisage++*, users can select desired x-ranges to perform matching through brushing interactions (Figure 2B2) or enter a y-range selection in the filter constraint textbox (Figure 2D4). Other interfaces for range selection in past VQSs include textboxes [?, ?], min/max line boundaries [?], or brushing interactions [?]. TimeSearcher and Queryline's approach is most flexible for range selection as they allow composition of multiple ranges to formulate complex piecewise queries, such as finding gene expression profiles rising from  $x=1-5$  then declining from  $x=5-10$ . Inspired by TimeSearcher's brushing interaction, we support only brushing for  $x$ , since in most of our use cases it was more common to focus the context based on the independent variable, such as zooming into particular sharp dips when looking for planetary transits or anomalous peaks indicative of erroneous experimental measurements.

**F3:** In addition to controls for fine-tuning the portions of the sketch to be matched, VQSs also need to allow users to control the underlying matching algorithm. In *zenvisage++*, users have the option to change similarity metrics to perform flexible matching (Figure 2D1). *zenvisage++* supports an option to ignore the  $x$ -range in shape matching (Figure 2D4). This feature is similar to 'temporal invariants' in SketchQuery [?].

### III) View specification

**Purpose:** Modify visualization settings for all visualizations displayed on the VQS. The ability to change the view specification offers analysts different perspectives on the same portion of data.

#### Motivating Use Case Challenges:

- C1:** For non-time-series uses of VQS, such as in material science, analysts is often interested in pivoting across various different  $x$  and  $y$  axes attributes.
- C2:** Astronomers often view magnitude measurements by reversing the  $y$ -axes, material scientist often prefer viewing their data in terms of a scatterplot rather than line charts.

#### Instantiated Feature:

- F1:** The data selection panel in *zenvisage++* allows users to select dataset and attributes of interest.
- F2:** Display options can be modified via the control panel in *zenvisage++*, such as reversing the  $y$  axes or changing visualization mark type to scatter.

### IV) Slice-and-Dice

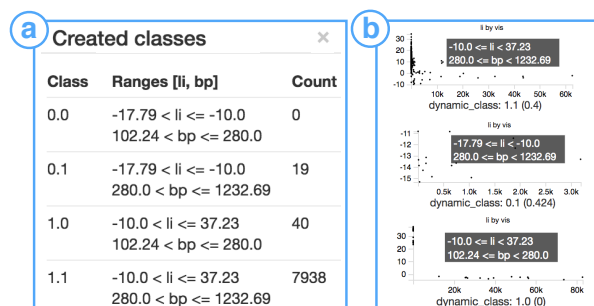
**Purpose:** Navigate and compare collections of visualizations constructed from different portions of the data.

#### Motivating Use Case Challenges:

- C1:** Since astronomers often have datasets with large numbers of objects, they first use domain knowledge to narrow down their search to a more manageable subset to increase their chances of finding an interesting pattern for a given pattern query.
- C2:** Material scientists often partition solvent datapoints into customized classes based on their physical properties, in order to compare between these classes. For example, M1 wanted to create classes of solvents with ionization potential under  $-10$  kJ/mol, over  $-8$  kJ/mol, and ones between that range, and examine how visualizations involving lithium solvation energy varied across the three classes.

#### Instantiated Feature:

- F1:** To filter data on-the-fly in *zenvisage++*, users can compose one or more conditions as filter constraints in a textbox (Figure 2D3). The filtering can be done on data columns associated with each pattern that is not visualized or on the visualized attributes. This feature is unique to *zenvisage++* as most existing VQSs do not allow users to interact with data in the non-visualized columns.
- F2:** *zenvisage++* supports dynamic class creation, a feature that allows users to create custom classes interactively, based on multiple data properties, following which each class can be visualized as a separate visualization (Figure 4). Information regarding the created classes is displayed in a table and as a tooltip over the aggregate visualizations.



**Figure 4:** 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.

### V) Result querying

**Purpose:** Querying based on the displayed results (either from the ranked list in Figure 2C or recommendations from Figure 2E), essentially requesting for patterns similar to the selected data pattern.

#### Motivating Use Case Challenges:

- C1:** General need for having to search for a pattern similar to a visualization of interest. For example, G1 was interested in the gene 'Esrrb' and wanted to find other genes that exhibits a similar pattern to it.

Process	Component				
	Top-Down	Context Creation	Bottom-Up	Match Specification	View Specification
TimeSearcher [8,9]	✓	✓	✓	✓	✓
QuerySketch [24]	✓	✓	✓	✓	✓
QueryLines [20]	✓	✓	✓	✓	✓
SoftSelect [10]	✓	✓	✓	✓	✓
Google Correlate [15]	✓	✓	✓	✓	✓
TimeSketch [5]	✓	✓	✓	✓	✓
SketchQuery [3]	✓	✓	✓	✓	✓
Qetch [13]	✓	✓	✓	✓	✓
Zenvisage [22,23]	✓	✓	✓	✓	✓
Zenvisage ++	✓	✓	✓	✓	✓

Table 2: Table summarizing whether key functionalities of VQSs (columns) are covered by past systems (row), indicated by checked cells. Column header colors blue, orange, green represents three sensemaking process (top-down querying, search with context, and bottom-up querying) described in Section 4.2. The heavily-used, practical features in our study for context-creation and bottom-up inquiry is largely missing from prior VQSs.

#### Instantiated Feature:

**F1:** In *zenvisage++*, users can drag and drop a visualization in either the results pane or the representative and outliers to the query canvas (Figure 2j). Similarly, TimeSearcher enable users to instantiate queries via drag-and-drop, whereas QuerySketch does so through double clicking.

#### VI) Recommendation

**Purpose:** displays visualizations that may be of interest to the users based on the data context.

#### Motivating Use Case Challenges:

**C1:** Geneticists are often interested in learning about the common gene expression profiles in their dataset.

#### Instantiated Feature:

**F1:** *zenvisage++* provides visualizations of representative trends based on clustering and highlights outlier instances (Figure 2E).

### 4.3. 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 5 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 [?], 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 5 blue) in a fixed collection. On the other hand, in the realm of *bottom-up data-driven inquiry* (Figure 5 green), the pattern of interest is unknown and 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 5 yellow)—i.e., setting the stage for bottom-up or top-down processes—*Doris: this clause is awkward?* is essential for allowing users to pivot across different visualization collections.

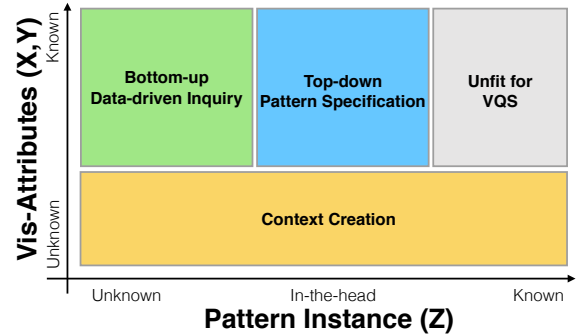


Figure 5: The problem space for VQSs is characterized by how much the analyst knows about the visualized attributes and the pattern instance. Colored areas highlight the three sensemaking processes in VQSs for addressing these characteristic problems. While prior work has focused solely on use cases 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.

### 4.4. Design Goals for the Sensemaking Processes

After understanding how each sensemaking process fits into the problem space addressed 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

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 taxonomy in Figure 3, 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 jump-start 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 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 [2, ?]. 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 accelerate the process of visualization discovery.**

## 5. 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. 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.

### 5.1. 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 their pattern onto the canvas (Figure 2a). The main reason why participants did not find sketching that 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 sketch canvas via an action like Figure 2j). 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.

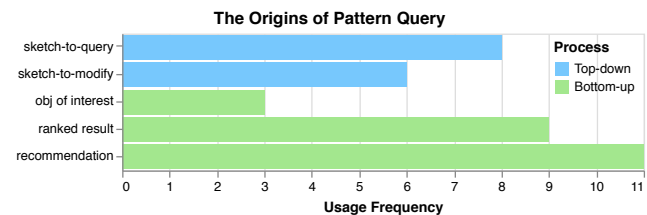


Figure 6: The number of times a pattern query originates from one of the workflows. We find that pattern queries are more commonly generated via bottom-up than top-down processes.

Given these initial findings, we further investigated where the pattern on the canvas typically originates, as presented in Figure 6. 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 7 (left). M2 repeated this workflow twice in separate occurrences during the study and was able to derive insights from the results. Likewise, Figure 7 (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 6 shows that bottom-up processes are more common than top-down processes for generating a pattern query.

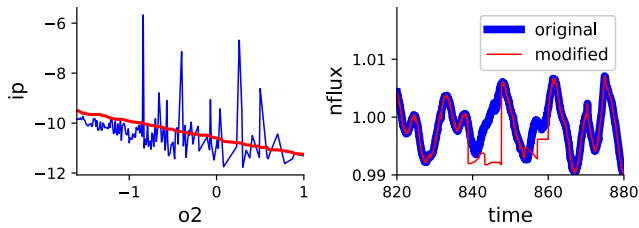


Figure 7: Canvas traces from M2 (left) and A3 (right) during the study demonstrating query modification. The original drag-and-dropped query is shown in blue and the sketch-modified queries in red.

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 an prescriptive, ad-hoc 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 6, we can see that sketch-to-query was only used 8 times, while the remaining modes 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 2 shows 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.

## 5.2. Context Creation and Bottom-up Approaches

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<sup>‡</sup>. We now highlight some of the exemplary workflows demonstrating the efficacy of the latter two sensemaking processes.

As shown in Figure 6, the most common use of bottom-up querying is via recommended visualizations. For example, G2 and G3

<sup>‡</sup> Feature usefulness is computed based on a matrix of usefulness labels for each use case (defined by a feature and user pair). The percentage of useful features for each process is computed as: 
$$\frac{\text{\# of useful features in process}}{\text{total \# of features in process} \times \text{total \# of users}}$$

identified that the three representative patterns recommended in *zenvisage++* corresponded to the same three groups of genes discussed in a recent publication [?]: induced genes (profiles with expression levels staying up), repressed genes (started high but went down), and transients (go up and then come down at different time points). 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 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 [?, ?]. 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 6. 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 and 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!

### 5.3. The Sensemaking Process in VQSs

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 Figure 3, 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 8) 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.

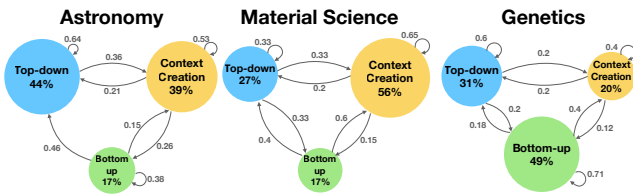


Figure 8: Markov models computed based on the evaluation study event sequences, with edges denoting the probability that a participant in the particular domain will go from one sensemaking process to the next. Nodes are scaled according to the eigenvector centrality, which represents the percentage of time users would spend in a particular 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 8. 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. Given that nodes in Figure 8 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. For example, without an initial query in-the-head, geneticists relied heavily on bottom-up querying through recommendations to jumpstart their queries.

The 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 8 highlight how the three sensemaking processes do not simply follow a linear progression, going from unknown to known pattern instance and visualized attributes in the problem space. Instead, the high connectivity of the transition model illustrates how these three equally-important processes form a sensemaking loop. The VQS sensemaking loop represents iterative acts of dynamic foraging and hypothesis generation. This flexibility is enabled by the diverse set of potential workflows that could be constructed in a full-fledged VQS like *zenvisage++*, for addressing a wide range of analytical inquiries.

### 5.4. 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 8), 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.

### 6. 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.