

# *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 various 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, visualization, exploratory data analysis, visual query, scientific data.

## 1 INTRODUCTION

Two-dimensional line charts are one of the most ubiquitous visualization types, due to how the simple and intuitive patterns often illustrate complex underlying processes and narrate interpretable and visually-compelling data-driven stories. To discover these patterns, analysts currently create visualizations, either programmatically using tools (such as `ggplot` or `matplotlib`), or visualization construction interfaces (such as Excel or Tableau) by 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, scientists need to individually inspect the visualizations of each object (often numbering in the thousands) until they find ones that match the pattern. This process of manually exploring large numbers of visualizations is not only error-prone, but also overwhelming for analysts who do not have extensive knowledge about their dataset.

To address this challenge, there has been a large number of work dedicated to building *Visual Query Systems* (VQSs), that allow users to specify desired visual patterns via an interactive interface [3, 5, 9, 10, 13, 15, 20, 22, 24]. Most of these systems has a sketching interface for drawing a trend of interest, with the system automatically traversing all potential visualization candidates to find those that match the specification. Since sketches can be ambiguous, many have also developed finer-grained specification techniques for enabling users to clarify how a sketch should be interpreted [3, 5, 10, 13, 20].

While these intuitive interface seem to be the 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 between current research to understand how VQSs can actually be used in practice, as a first step towards the broad adoption of VQSs in data analysis.* In this paper, we present findings from a series of interviews, cognitive walkthroughs, participatory design, and user studies with scientists from three different scientific domains—*astronomy*, *genetics*, and *material science*—through a year-long collaboration. These scientific use cases represent a diverse set of goals and datasets where 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 learned about the challenges in participant's existing data analysis workflows that could be potentially addressed by a VQS, described in Section 3. Building on top of the existing, open-source VQS, *zenvisage* [22, 23], we collaborated closely with scientists to gather feedback and iterate on VQS feature designs. The features we developed are organized into a taxonomy of functionalities in VQSs. The taxonomy introduces three sensemaking processes spanning over the problem space for VQSs. We find that prior VQS have focussed largely on top-down processes, such as mechanisms for specifying a pattern or how matching should be done, while largely missing out on two sensemaking process (context creation and bottom-up inquiries) that are crucial for the needs of the analysts in our study.

To study how various VQSs functionalities are used in practice, we conducted a user study with nine scientists 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 how sketching is impractical in most use cases. This is in part due to how top-down processes, such as sketching, adopts a problematic assumption that analysts start with a *known* and *easy-to-specify* search pattern in mind. In practice, users may not always know what pattern they want to sketch, nor are the patterns easily sketchable. While it is often easy for analysts to interpret a visualization once they see it, coming up with a pattern to query in an ad-hoc manner can often be challenging. We suspect this is why sketch-based systems have not been widely adopted in practice, pointing to the need for two other sensemaking processes, described more in Section 5.

Further analysis on how participants transition between different sensemaking processes illustrated how participants adopt a diverse set of workflows tailored to their problem contexts. For example, participants often construct a central workflow around a main sensemaking process, while interleaving variations with the two other processes as they iterate on an analytic task. This points to how all three sensemaking processes are essential for enabling users to transition smoothly between different workflows within the sensemaking loop. Our contributions include:

- domain problem characterization of visual querying through design studies with three different subject areas,
- abstraction of taxonomy and problem space of VQSs grounded in participatory design findings,
- a full-fledge VQS, *zenvisage++*, capable of facilitating rapid hypothesis generation and insight discovery,
- evaluation study findings regarding how VQSs are used in practice, leading to the formation of a novel sensemaking model for VQSs.

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 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

### 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 [8, 9], where the query specification mechanism is a rectangular box, filtering out all of the time series that does not pass through it, QuerySketch [24] and Google Correlate [15], 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 [3, 5, 13] and improving the expressiveness of sketched queries through finer-grained specification interfaces and pattern-matching algorithms [10, 20].

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 [3, 9], 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 [16], 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 [12, 14, 21] with three different subject areas for domain problem characterization. Comparing and contrasting between the diverse set of questions, datasets, and challenges from each use case enabled us to better characterize the problem to more generalized VQS use cases.** Based on these findings, we develop a feature taxonomy for understanding the sensemaking process in VQSs as part of *data/operation abstraction design*. Finally, we validated the abstraction design with grounded evaluation [11, 19], 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.**

### 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 from three research groups participated in the design of *zenvisage*. On average, participants had more than 8 years of research experience working in their respective fields. Via interviews and cognitive walkthroughs with researchers from the three different scientific research groups, we identified the needs and challenges of these use cases.

For the participatory design study, we built on an existing VQS, *zenvisage* [22, 23], that allowed users to sketch a pattern or drag-and-drop an existing visualization as a query, with the

	Pattern Specification	Match Specification	View Specification	Slice-and-Dice	Result Querying	Recommendation
TimeSearcher [8,9]			✓	✓		✓
QuerySketch [24]	✓	✓	✓			
QueryLines [20]	✓	✓	✓			
SoftSelect [10]	✓	✓	✓			
Google Correlate [15]	✓	✓	✓			
TimeSketch [5]	✓	✓	✓			
SketchQuery [3]	✓	✓	✓		✓	
Qetch [13]	✓	✓	✓			
Zenvisage [22,23]	✓	✓	✓		✓	✓
Zenvisage ++	✓	✓	✓	✓	✓	✓

**Table 1: 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. The heavily-used, practical features in our study for context-creation and bottom-up inquiry is largely missing from prior VQSs.**

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 1).

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 how VQSs could help address their research questions. 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.**

### 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 2**, the evaluation study participants included the six scientists from participatory design, along with three additional “blank-slate” participants who had never encountered *zenvisage++* before. While

participatory design subjects actively provided feedback on *zenvisage++* with their data, they only saw us demonstrating their requested features and explaining the system to them, rather than actively using the system on their own. So the evaluation study was the first time that all participants used *zenvisage++* to explore their datasets.

Participants for the evaluation study 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. 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 the system details and given approximately ten minutes to experience a guided exploration of *zenvisage++* with a preloaded real-estate example dataset from Zillow [1]. 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.

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

## 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.

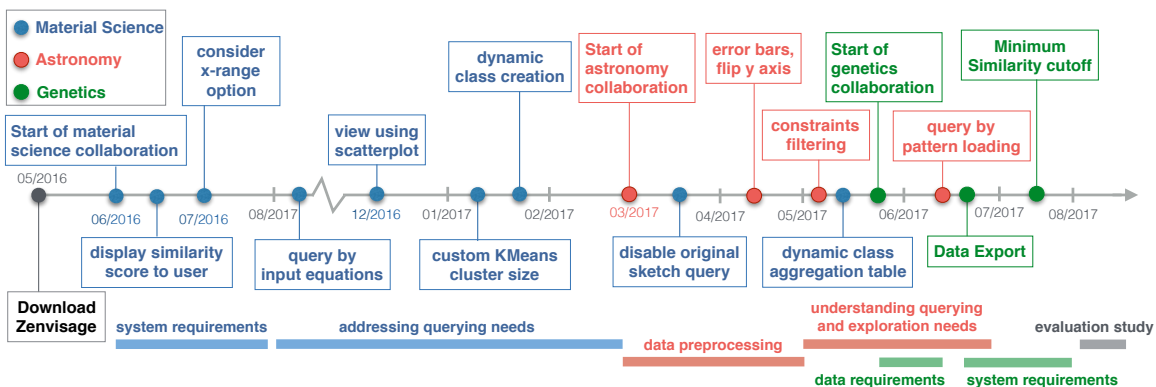


Figure 1: Timeline for progress in participatory design studies.

**Astronomy:** The Dark Energy Survey is a multi-institution project that surveys 300 million galaxies over 525 nights to study dark energy [4]. The telescope *used to survey these galaxies* also focuses on smaller patches of the sky on a weekly interval to discover astrophysical 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 astrophysical object, called a *light 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 astrophysical 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 [6]. 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

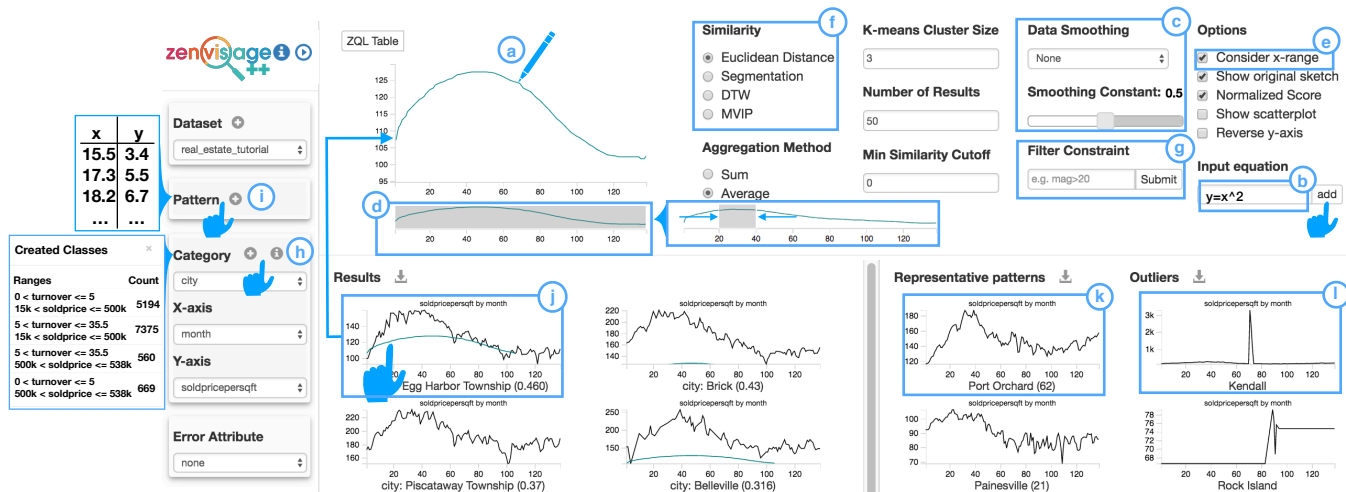
looks “clean” and checks that the number of outlier genes (i.e., those that do not fall into any of the clusters) is low. If the number of outliers is high or the clustered visualizations look “unclean”, she reruns the analysis by increasing the number of clusters. When the visualized clusters look “good enough”, G1 exports the cluster patterns 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 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. 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





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

impossible for them to uncover hidden relationships between different attributes across large number of solvents manually.

#### 4 PARTICIPATORY DESIGN FINDINGS

Given that participants from all of the aforementioned domains recognized the need for a tool for visual pattern search, we worked closely with them to develop features to address their characteristic problems and challenges. Next, we describe features that we incorporated into our enhanced VQS, *zenvisage++*, thematically organized by component. Then, we introduce a taxonomy for organizing these feature components into three sensemaking processes, spanning different problems that VQSs are aimed to solve.

##### Themes Emerging from Participatory Design

**Pattern Specification** interfaces allow users to submit exact descriptions of a pattern as a query (hereafter referred to as *pattern query*), with the VQS returning a list of most similar matches. Almost all existing VQSs support freehand sketching for specifying desired patterns (Figure 2a). Since it is often difficult to sketch precisely, *zenvisage++* also allows users to specify a functional form (e.g.,  $y=x^2$ ) for a pattern (Figure 2b). This feature was requested by material scientists who were interested in finding solvents with known analytical models describing chemical relationships. *zenvisage++* also enables users to upload a pattern consisting of a sequence of points as a query (Figure 2i). The uploaded pattern also represents a more precise query specification that captures the desired features of a pattern that cannot be precisely sketched. For example, the width of a supernovae light curve is characteristic to the radioactive decay rate of its chemical signature [17], so querying with an exact pattern template would be helpful for

distinguishing the patterns of interest from noise. The pattern upload functionality is also available in Google Correlate.

**Match Specification:** Past work has shown that pattern queries can be extremely imprecise [3, 5, 10]. To this end, VQSs need to support mechanisms for clarifying sketch interpretation (i.e., how matching should be performed). Motivated by the dense and noisy observational data in astronomy and material science, we developed an interface for users to interactively adjust smoothing algorithms and parameters on-the-fly to update the resulting visualizations accordingly (Figure 2c). *zenvisage++* support data smoothing to allow users to interactively change the degree of shape approximation they would like to apply to all visualizations (and consequently for pattern matching). Smoothing is also supported in Qetch. Other interfaces have developed constrained sketching mechanisms to allow users to partially specify certain shape characteristics, such as angular slope queries [9] or piecewise trend querylines [20].

Time series analysis often involves specific ranges of interest with special domain significance. To find such patterns in *zenvisage++*, users can limit pattern queries to be matched only in specific x or y ranges, specified through brushing mechanism to select desirable x-ranges to perform shape matching (Figure 2d) and filter textbox entry for y-range selection. Other interfaces for range selection in past systems includes textboxes [13, 24], min/max line boundaries [20], or brushing interactions [8]. 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  $x=5-10$ .

In addition to controls for enriching how portions of the sketch should be interpreted, VQSs also need to enable users

to control the underlying matching algorithm. In *zenvisage++*, users have the option to change similarity metrics that perform flexible matching (Figure 2e). In the astronomy and genetics use case, the participants were interested in patterns, such as the existence of a peak or a rising profile, without regards to the exact time when the event occurs. Similar to temporal invariants in SketchQuery, *zenvisage++* supports an option to ignore the x-range in shape matching (Figure 2f).

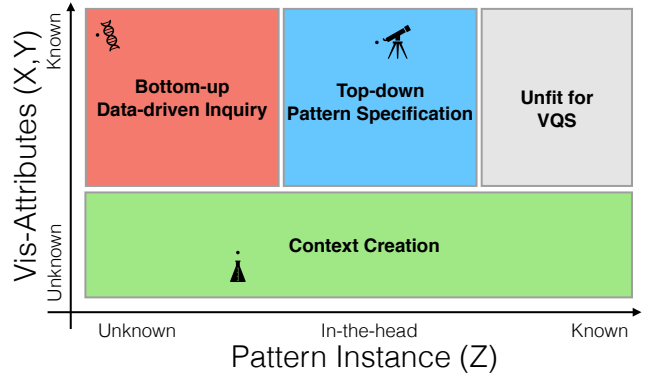
**View specification** interfaces are settings that alters the visualization specification of all visualizations displayed on the VQS. These include changing the visualized attributes on the x and y axes, as well as display options, such as reversing the y axes (common operation done by astronomers visualizing magnitude measurements) and changing visualization mark type as scatter (material science dataset represents each solvent as a datapoint better visualized as a scatterplot). The ability to change view specification offers users different perspectives on the same portion of data.

**Slice-and-Dice** empowers users to navigate and compare different collections of visualizations constructed from different regions of the data. Since astronomers often have datasets with large numbers of objects, they need to first use domain knowledge to narrow down their search to a more manageable subset of data. This increases their chances of finding an interesting pattern for a given pattern query. To filter data on-the-fly in *zenvisage++*, users could compose one or more conditions as filter constraints in a textbox (Figure 2g).

Another common slice-and-dice workflow is bucketing data points into customized classes based on existing properties, then 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. To this end, we implemented dynamic class creation, a feature that allows users to create custom classes interactively, based on multiple data properties (Figure 2h). Information regarding the created classes is displayed in a table and as a tooltip over aggregate visualizations.

**Result querying** allows users to submit a query based on the results, essentially asking for patterns similar to the selected data pattern. 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.

**Recommendation** displays visualizations that may be of interest to the users based on the data context. *zenvisage* provides visualizations of representative trends based on clustering and highlights outlier instances (Figure 2k,l).



**Figure 3: The problem space of VQSs is characterized by how much the analyst knows about the visualized attributes and pattern instance. Colored area highlights the three different paradigms of VQSs. 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 coverage in the red and green regions.**

### Characterizing Design Space for VQSs

Given the participatory design results, we further characterize three sensemaking process in the problem space of VQSs. Visual querying often consists of searching for a desired visualization instance ( $Z$ ) across a visualization collection that consists of some attributes ( $X,Y$ ). 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 contain 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) (e.g. only interested in patterns related to a specific gene), visualization-at-a-time system, 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 [18], 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). On the other hand, in the realm of *bottom-up data-driven inquiry* (Figure 3 red), the pattern of interest is unbeknownst and external to the user and must be driven by recommendations or queries that originate from the data. As we will discuss later, this process is a crucial but understudied topic in past works on VQSs.

The second axis, **visualized attributes**, depicts how much the analyst knows about which  $X$  and  $Y$  axes she is interested

Taxonomy of Functionalities in Visual Query Systems

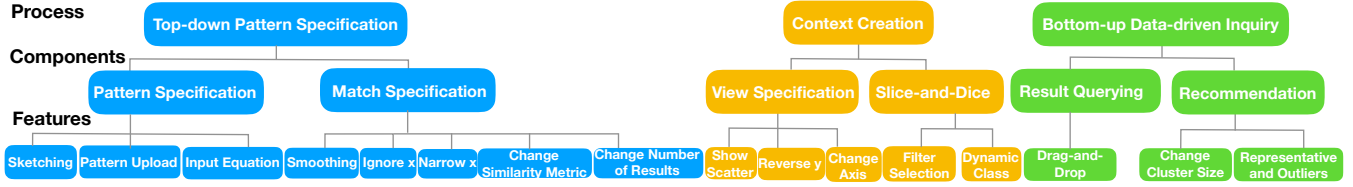


Figure 4: Taxonomy of functionalities in VQSs. From top, each of the three paradigm is broken down into key components in the system, which is instantiated as features in *zenvisage*. The bottom-most layer connects the use cases features that have practical or envisioned usage based on the evaluation study.

in visualizing. In both the astronomy and genetics use cases, as well as past work in this space, data was in the form of 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 the realm of unknown attributes, context creation (Figure 3 green) is essential for allowing users to pivot across different visualization collections.

### Design Goals and Challenges for VQS Paradigms

We further explore the design objectives and challenges in supporting each sensemaking process by developing a taxonomy for organizing the aforementioned components.

**Top-down Pattern Specification** begins with user’s intuition about how their desired patterns should look like based on ‘theory’, including visualizations from past experiences or abstract conceptions based on external knowledge. The goal of top-down pattern specification is to address the *which* questions 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 browse-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. They queried mainly through these recommended results 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 and to learn about *where the patterns of interest lies*. 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 create different dynamic classes to study their data from different 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.

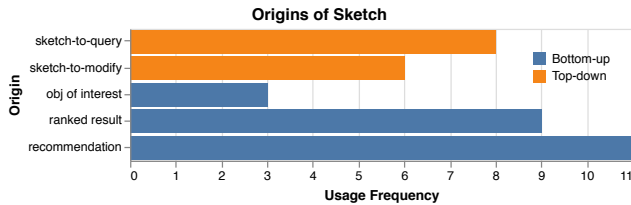
## 5 EVALUATION STUDY FINDINGS

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 regarding a system feature, request for functionalities unaddressed by the system, and the use of external tools. To characterize the usefulness of each feature, we further labeled whether each feature was useful to a particular user’s analysis. We regard a feature as *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. Next, we will make use of the results from this thematic analysis to understand how feature usage informs the roles of each sensemaking paradigms in real-world analytic tasks. Aditya: Can’t parse the previous sentence

## 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 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.

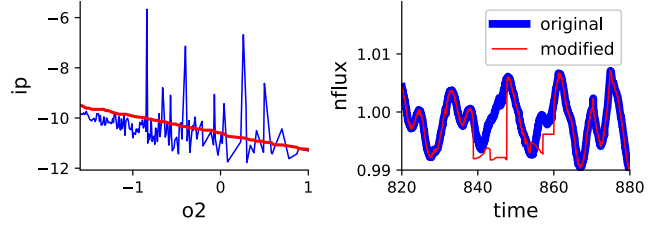
Given these initial findings, we further investigated where the pattern on the canvas typically originates. Figure 5 shows that pattern queries originate from either top-down (sketching) or bottom-up (drag-and-drop) approaches. *Aditya: We don't really define what the various origins are clearly: what is sketch-to-query? what is sketch-to-modify?. Always describe what is going on in a figure clearly. I am left confused as I read through the rest. We shouldn't call it a sketch – let's call it a pattern query. Delete the title from figure 5 Aditya: got until here. for any changes for the previous text, please do it in color.* Within these actions, there can be different intentions behind the sketch. While all visualizations that could be drag-and-dropped must come from the result or recommendation pane, a query can come from a particular object that the participant is interested in or simply through peripheral browsing of visualization results.



**Figure 5: The number of times each pattern query is generated by one of the workflows.**

There are also many unexpected use cases where sketching was simply used as a mechanism to modify an existing pattern query. 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 noisy trend by tracing a sketch over the visualization, 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).



**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 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 users queried by equation. In both astronomy and genetics, the visualization patterns result 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.

These findings suggest that while sketching is an useful analogy for people to express their queries, *the existing ad-hoc, sketch-only model for visualization querying is insufficient without data examples that can help analysts jumpstart their exploration.* This finding has profound implications on the design of future VQs, since Table 1 show that past work have primarily focused on optimizing components in the top-down paradigm, without regards to how useful these features are in real-world analytic tasks. We suspect that these limitations may be why existing VQs are not commonly adopted in practice.

## Context Creation and Bottom-up Approaches

*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. We find that top-down approaches was only useful 29% of the use cases, whereas it was useful for 70% of the use cases for bottom-up approaches and 67% for context creation.

As shown in Figure 5, the most common use of bottom-up querying is via recommended visualizations. Examples of how recommended trends can provoke further insightful actions comes from G2 and G3, who identified that the three representative patterns shown in *zenvisage++*—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)—corresponded to the same three groups of genes discussed in a recent publication [6]. 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 cluster 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. G3 described the process of using VQSs as doing “detective work” that provoked him to generate further scientific hypotheses as well as data actions. By browsing through the ranked list of result 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 [2, 7]. Context creation in VQSs enables users to change the lens in which they look through when performing visual querying, thereby creating more opportunities to see the queried 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 to

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:

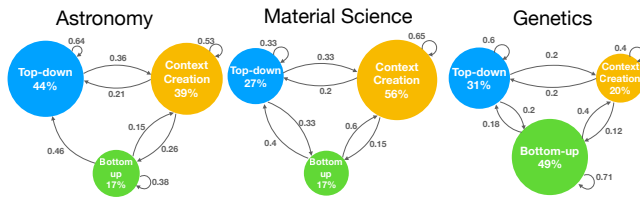
All this is really possible because of dynamic class creation, so this allows you to bucket your intuition and put that together. [...] I can now bucket things as high voltage stable, liquid stable, viscous, or not viscous and start doing this classification quickly and start to explore trends. [...]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

### The Sensemaking Process in VQSs

Given our observation on how analysts make use of each sensemaking process in practice, we further investigate the interplay of 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 4, we map each of the feature usage to one of the three sensemaking processes. Each participant’s event sequence is divided into sessions indicating a separate lines of inquiry during the analysis. Based on the event sequence for each session, we compute the state transition probabilities (shown as edge weights in Figure 7) to characterize how analysts 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 analyst would continue with the same type of sensemaking process. For example, when an astronomer performs top-down pattern specification, it is proceeded 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.

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 users spend in each of the sensemaking processes when the transition model has evolved to a steady state. Given that nodes in Figure 7 are scaled by this value, in all subject areas, we observe that there is always a prominent node connected to two other smaller nodes. Our observation demonstrates how participants often construct a central workflow around a main sensemaking process and interleave variations with the two



**Figure 7: Markov model computed based on evaluation study event sequences, with edges denoting the probability that an analyst in the particular subject area will go from one sense-making process to the next. Nodes are scaled according to the eigenvector centrality, which represents the percentage of time users spend in a particular state.**

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 paradigm adopted by each use case is tightly coupled with characteristics of the analytic challenges presented by each 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 graph highlights how the three sensemaking process does 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.

## Limitations

Although evidence from our evaluation study suggests that 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 use cases, 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 includes: financial data for business intelligence, electronic medical records for healthcare,

and personal data for “Quantified Self”. These different use cases may each pose different sets of challenges unaddressed by the findings in this paper, pointing to a promising direction for future work.

## REFERENCES

- [1] 2016. Zillow. [www.zillow.com](http://www.zillow.com). Accessed: February 1, 2016.
- [2] Robert Amar, James Eagan, and John Stasko. 2005. Low-level components of analytic activity in information visualization. In *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*. IEEE, 111–117. <https://doi.org/10.1109/INFOVIS.2005.24>
- [3] Michael Correll and Michael Gleicher. 2016. The semantics of sketch: Flexibility in visual query systems for time series data. In *Visual Analytics Science and Technology (VAST), 2016 IEEE Conference on*. IEEE, 131–140. <https://doi.org/10.1109/VAST.2016.7883519>
- [4] Drlica Wagner et al. 2017. Dark Energy Survey Year 1 Results: Photometric Data Set for Cosmology. (2017). arXiv:1708.01531
- [5] Philipp Eichmann and Emanuel Zgraggen. 2015. 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. <https://doi.org/10.1145/2678025.2701379>
- [6] Gloss et al. 2017. High resolution temporal transcriptomics of mouse embryoid body development reveals complex expression dynamics of coding and noncoding loci. *Scientific Reports* 7, 1 (2017), 6731. <https://doi.org/10.1038/s41598-017-06110-5>
- [7] Jeffrey Heer and Ben Shneiderman. 2012. A taxonomy of tools that support the fluent and flexible use of visualizations. *Interactive Dynamics for Visual Analysis* 10 (2012), 1–26. <https://doi.org/10.1145/2133416.2146416>
- [8] Harry Hochheiser and Ben Shneiderman. 2001. Interactive Exploration of Time Series Data. In *Discovery Science*. Springer, Berlin, Heidelberg, 441–446.
- [9] Harry Hochheiser and Ben Shneiderman. 2004. Dynamic query tools for time series data sets: Timebox widgets for interactive exploration. *Information Visualization* 3, 1 (2004), 1–18.
- [10] Christian Holz and Steven Feiner. 2009. Relaxed Selection Techniques for Querying Time-series Graphs. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology (UIST '09)*. ACM, New York, NY, USA, 213–222. <https://doi.org/10.1145/1622176.1622217>
- [11] Petra Isenberg, Torre Zuk, Christopher Collins, and Sheelagh Carpendale. 2008. 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. <https://doi.org/10.1145/1377966.1377974>
- [12] Heidi Lam, Enrico Bertini, Petra Isenberg, Catherine Plaisant, and Sheelagh Carpendale. 2012. Empirical studies in information visualization: Seven scenarios. *IEEE Transactions on Visualization and Computer Graphics* 18, 9 (2012), 1520–1536. <https://doi.org/10.1109/TVCG.2011.279>
- [13] Miro Mannino and Azza Abouzied. 2018. Expressive Time Series Querying with Hand-Drawn Scale-Free Sketches. (2018), 1–12. <https://doi.org/10.1145/3173574.3173962>
- [14] Tamara Munzner Michael Sedlmair, Miriah Meyer. 2012. Design Study Methodology: Reflections from the Trenches and the Stacks. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (Dec 2012), 2431–2440. <https://doi.org/10.1109/TVCG.2012.213>

- [15] Matt Mohebbi, Dan Vanderkam, Julia Kodysh, Rob Schonberger, Hyun-young Choi, and Sanjiv Kumar. 2011. Google correlate whitepaper. (2011).
- [16] Tamara Munzner. 2009. A nested model for visualization design and validation. *IEEE transactions on visualization and computer graphics* 15, 6 (2009). <https://doi.org/10.1109/TVCG.2009.111>
- [17] Nugent et al. 1997. Synthetic Spectra of Hydrodynamic Models of Type Ia Supernovae. *The Astrophysical Journal* 485, 2 (1997), 812–819. <https://doi.org/10.1086/304459>
- [18] Peter Pirolli and Stuart Card. 2005. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of International Conference on Intelligence Analysis*, Vol. 5. 2–4.
- [19] Catherine Plaisant. 2004. The challenge of information visualization evaluation. In *Proceedings of the Working Conference on Advanced Visual Interfaces*. ACM, 109–116. <https://doi.org/10.1145/989863.989880>
- [20] Ryall et al. 2005. Querylines: approximate query for visual browsing. In *CHI'05 Extended Abstracts on Human Factors in Computing Systems*. ACM, 1765–1768. <https://doi.org/10.1145/1056808.1057017>
- [21] Ben Shneiderman and Catherine Plaisant. 2006. 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*. ACM, 1–7. <https://doi.org/10.1145/1168149.1168158>
- [22] Tarique Siddiqui, Albert Kim, John Lee, Karrie Karahalios, and Aditya Parameswaran. 2016. Effortless data exploration with zenvisage: an expressive and interactive visual analytics system. *Proceedings of the VLDB Endowment* 10, 4 (2016), 457–468. <https://doi.org/10.14778/3025111.3025126>
- [23] Siddiqui et al. 2017. Fast-Forwarding to Desired Visualizations with Zenvisage. In *CIDR*.
- [24] Martin Wattenberg. 2001. Sketching a graph to query a time-series database. In *CHI'01 Extended Abstracts on Human factors in Computing Systems*. ACM, 381–382. <https://doi.org/10.1145/634067.634292>