

How are visual query systems used in practice: A Design Study with Zenvisage

ABSTRACT

Visual query systems (VQSs) allow users to interactively search for time-series with desired visual patterns using intuitive sketch-based interfaces. Despite their supposed promise, more than a decade of past work on VQSs has not been translated to practical adoption. Through a year-long collaboration with scientists from three diverse domains, we study the types of problems posed to VQSs, develop features to support these use cases, and evaluate how VQS functionalities are used in practice. Through these observations, we formalize a taxonomy of key functionalities for VQSs to better understand how analysts perform sensemaking in such systems. To our surprise, we find that ad-hoc querying by sketching is unsuitable for exploratory analysis, since analysts do not often know what patterns they may be interested in. We advocate that next-generation VQSs should support two other forms of sensemaking to pave way for these systems to be adopted to 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, not only for time series but also for understanding the relationships between multiple measures variables.

From high-throughput genome sequencing, to multi-resolution astronomical imaging telescopes, to at-scale physical testing of battery candidates, many fields of science and engineering are facing an increasing availability of large volumes of complex data [4, 8], holding the key to some of the most pressing unanswered scientific questions of our time, 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? While data analysis is central to a scientist's knowledge discovery process, scientists often lack the extensive experience to deal with data of this scale and complexity in a way that can facilitate rapid insight discovery [15].

To explore their data, many scientists currently create visualizations, either programmatically using tools (such as

ggplot or matplotlib), or visualization construction interfaces (such as Excel or Tableau) [20, 25]. In either case, scientists are required to specify 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. Similarly, when trying to infer relationships between two physical properties for different subsets of battery electrolytes, scientists need to individually visualize these properties for each subset (out of an unbounded number of such subsets) until they identify relationships that make sense to them. This process of manually exploring a large number of visualizations is not only error-prone, but also overwhelming for scientists who do not have extensive knowledge about their dataset.

One potential solution for this challenge of manually exploring a large collection of visualizations are systems that allow users to specify desired visual patterns, via a high-level specification language or interface, with the system automatically traversing all potential visualization candidates to find and return those that match the specification. We define such systems to be *Visual Query Systems*, or VQSs for short. There are a number of VQSs that have been introduced in the literature [13, 19, 26, 29, 30]. For example, Google Correlate [19] and QuerySketch [30] allow users to draw a trend of interest, with the system automating the search to find matching visualizations.

Not surprisingly, when asked about the potential viability of VQSs in aiding scientific data analysis, many scientists indicated that VQSs could be useful in mitigating the challenge of visualization exploration described earlier—since it automates the painful manual exploration of visualizations to find desired patterns. Yet, to the best of our knowledge, VQSs are not very commonly used in practice. *Our paper seeks to bridge this gap between current research on VQSs to understand why existing VQSs are not commonly used by analysts and how they 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 present a diverse set of goals

and datasets that could benefit from a VQS. Our three main research questions are as follows:

RQ1: What are the challenges in existing scientific data analysis workflows that could be potentially addressed by a VQS?

Via cognitive walkthroughs and interviews, we gained an understanding of the data analysis workflows presently employed by the scientists, their needs, and the challenges they face. We identified opportunities where a VQS could help accelerate their analysis, by helping them discover insights, gain intuition, or provoke directions for exploration. Finally, we determined the types of research questions and dataset properties that would be most suitable for exploration on VQSs.

RQ2: What types of interface capabilities are necessary to develop VQSs into a useful component of data analysis?

Via participatory design, we distilled a set of key features that our scientist participants would need for VQSs to be useful and usable within their data analysis workflows. Based on our early interactions with scientists, we started to build a VQS [28, 29] that, similar to existing VQSs [30], allowed them to search for desired trends via drawing on a canvas. This early system served as a functional prototype for us to engage with scientists further in the participatory design process, understand how they envision themselves using a VQS, and gather feedback on feature designs that could make the VQS more useful. The features we developed address challenges shared across the three scientific domains, ranging from additional querying modalities, to features that support a more integrated workflow, to improving the interpretability of the system output, most of them missing in prior VQSs in the literature. Our collaborative design experience culminated in a full-fledged VQS, *zenvisage*, capable of facilitating rapid hypothesis generation and insight discovery.

RQ3: How do VQSs accelerate scientific insights? and RQ4: How can VQSs fit within the context of existing data analysis workflows?

To evaluate our final system *zenvisage*, we conducted a user study with nine scientists (including those who had participated in the design process), all of whom had a vested interest in using a VQS to address their research questions on their datasets. In a 1.5-hour user study, our scientist 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, finding characteristic gene expression profiles that confirmed the results of a related publication, and learning that the dip in an astronomical light curve is caused by saturated imaging equipment overlooked by the existing error-detection pipeline*. Participants also gained additional insights about their datasets, including debugging mislabelled features and uncovering the erroneous data preprocessing procedure applied to a collaborator’s dataset. We learned how VQSs could be contextualized within scientific data analysis

	Pattern Specification	Match Specification	View Specification	Slice-and-Dice	Result Querying	Recommendation
Timesearcher [12, 13]			✓	✓		✓
QuerySketch [30]		✓	✓			
QueryLines [26]		✓	✓			
SoftSelect [14]		✓	✓			
Google Correlate [19]		✓	✓			
TimeSketch [9]		✓	✓			
SketchQuery [7]		✓	✓			✓
Qetch [17]		✓	✓			✓
Zenvisage (prototype) [28]		✓	✓		✓	✓
Zenvisage (after design study)		✓	✓	✓	✓	✓

Table 1: Table summarizing the key components of VQSs (columns) covered by past systems (row). Green/red cell color indicates whether this feature exist in the system or not. Column header colors blue, orange, green represents processes: top-down querying, search with context, and bottom-up querying respectively, detailed in Section 4.

workflows and discovered that VQSs can be used beyond the exploratory phase of analysis, for data verification, debugging preliminary datasets, and performing sanity-checks on downstream models.

As most existing VQSs are evaluated in a standalone fashion via artificial tasks and datasets, to the best of our knowledge, *our study is the first to holistically examine how VQSs can be used in practice and integrated into existing data analysis workflows*. From these experiences, we advocate common design guidelines and end-user considerations for building the 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 of interest, including TimeSearcher [12, 13], where the query is composed of one or more rectangular value-range selections, and QuerySketch [30] and Google Correlate [19], where the query is sketched as a pattern. Subsequent work recognized the ambiguity of sketches and improved the expressiveness of sketched queries through finer specification interfaces and pattern-matching algorithms [14, 26], as well as performed crowdsourced perceptual studies to understand how humans rank similarity in patterns subjectively [7, 9, 17]. Table 1 summarizes the list of features offered by these existing systems.

Most of these systems have not been evaluated in-situ on real-world use cases. Even when design study was performed [7, 12], the focus these narrow use case in mind, and these — never adopted in practice, either — did not — didn’t influence the design decisions made to — . We found there is a genuine need in the community to more thoroughly understand the design space of VQSs and how various components of VQSs are used in practice. In this work, We make use of multiple case studies Participatory design has been successfully used in the development of interactive visualization systems in the past [3, 5]. Sedlmair et al. [18] advocate that design study methodology is suitable for use cases in which the data is available for prototyping, but the task is only partially known and the information is partially in the user’s head.

Participatory Design

Working with researchers from three different scientific research groups, we identified the needs and challenges of these use cases, via interviews and cognitive walkthroughs. We recruited participants by reaching out to research groups via email and word of mouth, who have experienced challenges in dealing with large amounts of data. 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. For the participatory design study, we built on an existing open-source VQS, *zenvisage*, that allowed users to sketch a pattern or drag-and-drop an existing visualization as a query, then the system would return visualizations that had the closest Euclidean distance from the queried pattern. The details of the system is described in [28, 29], which focused on the system and scalability aspects of the VQSs. Our three use cases were 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 8 years of research experience working in their respective fields.

During the participatory design process, we collaborated with each of the teams 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 engagement with the participants and the features inspired by their use cases can be found in Figure 1. Participants provided datasets they were exploring from their domain, whereby they had a vested interest in using a VQS to address their own research questions. Through this process, we identified and incorporated more than 20 desired features into our VQS prototype, *zenvisage*, over the period of a year.

Evaluation Study

In order to make the evaluation more realistic, at the end of our participatory design study, we opted for a qualitative evaluation where we invited participants to bring datasets that they have vested interests in to address unanswered research questions, in order to study how analysts interact with different VQS components in practice. 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 the potential participants to fill out a pre-study survey to determine their eligibility. Eligibility criteria included: being an active researcher in the subject area with more than one year of research experience, and having worked on a research project involving data of the same nature as that used in the participatory design. Four of the evaluation studies were conducted remotely. Participants had the option of exploring their own dataset or an existing dataset that they provided to us during the participatory design process. All three blank-slate participants opted to explore their own datasets.

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 our VQS 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 or use external tools as needed during the data exploration phase. If the participant was out of ideas, we suggested one of the ten main functionalities in *zenvisage* 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, the main exploration phase lasted for 63 minutes. After the study, we asked them open-ended questions about their experience.

3 PARTICIPANTS AND DATASETS

During the design study, we observed participants as they conducted a cognitive walkthrough demonstrating their existing data analysis workflow. In this section, we describe our study participants and their use cases to highlight behaviors

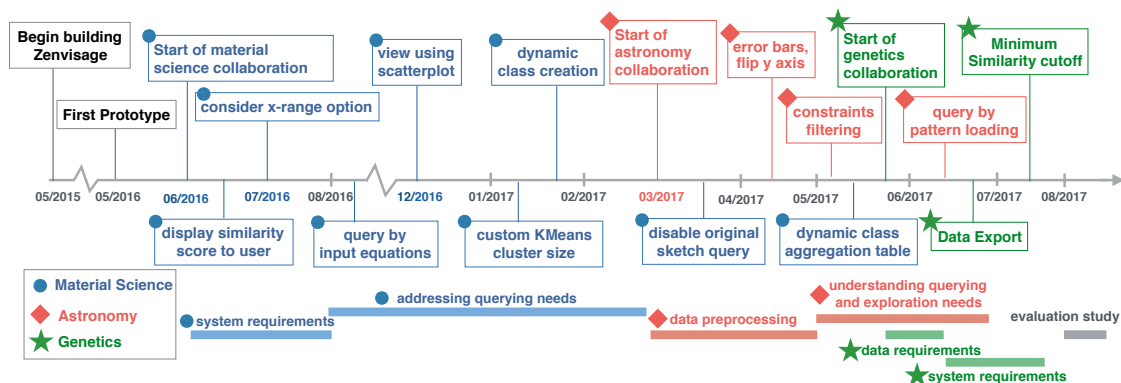


Figure 1: Participatory design timeline for the scientific use cases.

that participants have adopted for conducting certain analysis tasks.

Astronomy: The Dark Energy Survey (DES) is a multi-institution project that surveys 300 million galaxies over 525 nights to study dark energy. The telescope 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 supernova explosions or quasars. Their data consist of a large collection of time series brightness observations associated with each object. For over five months, we worked closely with A1, an astronomer on the project’s data management team working at a supercomputing facility. The scientific goal is to identify a smaller set of potential candidates that may be astrophysical transients in order to study their properties in more detail.

In order to identify astronomical 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 is time-consuming and error prone as the large majority of the objects are not astronomical transients. A1 was interested in *zenvisage* as he recognized how specific pattern queries could help scientists directly search for these rare objects.

Genetics: Gene expression is a common measurement in genomics 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 better understand how genes are related to phenotypes expressed during early development. Their data consisted of a collection of gene expression data over time for mouse stem cells aggregated over multiple experiments.

To analyze the data, G1 loads the preprocessed data into a custom desktop application for visualizing and clustering gene expression data. 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 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 be used as features in their downstream regression tasks.

Prior to the study, G1 and G3 spent over a month attempting to determine the best number of clusters for their upstream analysis 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 *zenvisage* as they saw how the ability to interactively query other 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 that can improve battery performance and stability. These scientists work with large simulation dataset containing chemical properties for more than 280,000 different solvents. 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 using VQSs. Each row of their dataset represents a unique solvent with 25 different chemical attributes. They wanted to use *zenvisage* 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 affects other properties under specific conditions.

M1 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 weigh in the cost and availability to determine experimental feasibility.

The scientists were interested in using *zenvisage* as it was impossible for them to uncover hidden relationships between different variables across large number of solvents manually.

4 PARTICIPATORY DESIGN FINDINGS

From participatory design, we learned about the characteristic problems and challenges present to VQSs. We first describe the features that we have developed to address these challenges, thematically organized by components. Along with analysis of past literature, we develop a taxonomy of key functionalities in VQSs. These components are then organized into three paradigms of sensemaking in VQSs that span across different areas in the design space.

Themes Emerging from Participatory Design

Pattern Specification interfaces allow users to submit an exact description of a pattern query, then the VQS returns a list of most similar matches. Almost all existing VQS supports freehand sketching for specifying desired patterns (Figure 2a). In addition to sketching, *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 relationships between their chemical properties. *zenvisage* and Google Correlate also enable users to upload a pattern consisting of a sequence of points as a query. The desired patterns to be uploaded are often associated with a concept, such as patterns generated from computational models or prelabelled data from an external reference database. For example, A1 wanted to query based on synthetic light curves generated from simulations and known supernovae that have been discovered in the past.

Match Specification: While exact shape specification is an intuitive mechanism for constructing a visual query, as pointed out by past works [7, 9, 14], pattern queries can be extremely imprecise. To this end, VQSs need to support mechanisms for clarifying sketch interpretation (i.e. how matching should be performed). Many interfaces have developed constrained sketching mechanism to allow users to partially specify certain shape characteristics, such as angular slope queries [13] or piecewise trend querylines [26]. Both Qetch and *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). Motivated by the dense and noisy observational data in the astronomy and material science use cases, we developed an interface for users to interactively adjust data smoothing algorithm and parameters on-the-fly to update the resulting visualizations accordingly (Figure 2c).

Another approach is to specify specific ranges where the matching should be performed. In time series analysis there are specific ranges of time and measure values with special

domain specific significance that may be of interest to users. To find such patterns, users can limit the pattern query to be matched only in specific x or y ranges, specified through textboxes [17, 30], min/max line boundaries [26], or brushing interactions [12]. *zenvisage* employs the brushing mechanism to select desirable x-ranges to perform shape matching (Figure 2d). Additionally, y axis range selection could be performed through entering a filter constraint on the measure variable. The TimeSearcher and Queryline 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 (Figure 2g), 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, so it is better visualized as a scatterplot). The ability to change view specification offers users different perspectives on the same portion of data.

Slice-and-Dice functionalities in VQSs provide a mechanism for navigating and comparing between different regions of the dataset, effectively empowering users to perform visual querying on different collections of visualizations constructed from the data subsets. To filter data on-the-fly in *zenvisage*, users could compose one or more conditions as filter constraints in a text field (Figure 2g). Users with large datasets often need to first use domain knowledge to narrow down their search to a subset of data. This increases their chances of finding an interesting pattern for a given pattern query.

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. Then, he wanted to see 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 use multiple properties to create custom classes

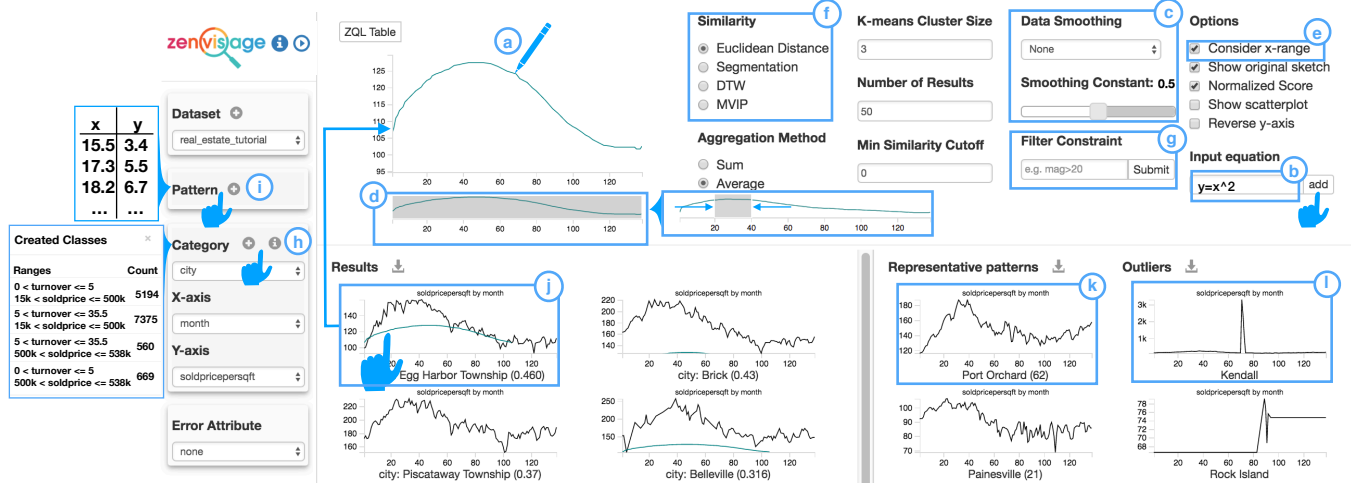


Figure 2: Our VQS after participatory design, which includes: the ability to query via (a) a sketch, (b) input equations, (i) drag and drop, or (j) uploaded patterns; (c) preprocessing via data smoothing; query specification mechanisms including (d) x-range selection and filtering, (e) x-range invariance, (g) Filtering, and (h) Dynamic class creation; recommendation of (k) representative and (l) outlier trends. Prior to the participatory design, *zenvisage* only included a single sketch input with no additional options.

on-the-fly, effectively slicing-and-dicing the data based on their needs (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 that are similar to the selected data pattern. TimeSearcher enable users to instantiate queries via drag-and-drop, whereas QuerySketch does so through double clicking. Similarly 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).

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).

Characterizing Design Space for VQSs

Based on example use cases and feature components from participatory design, we further characterize the design 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, as shown in Figure 3.

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), a user only be interested in patterns related to a specific gene. Such use cases would be more suited for a visualization-at-a-time system, where analyst manually

create and examine each visualization one at a time, rather than a VQS, since analysts can directly work with the selected instance without the need for visual querying. Inspired by Pirolli and Card’s information foraging framework [23], 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 (or equivalently, the visualization). As we will discuss latter, 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 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 subspaces.

Design Goals and Challenges for VQS Paradigms

We further explore the design objectives and challenges of each paradigm by developing a taxonomy for organizing how the aforementioned components fits into the paradigms of sensemaking in VQSs, as shown in Figure 4. In particular, we will describe the main form of inquiry addressed by each

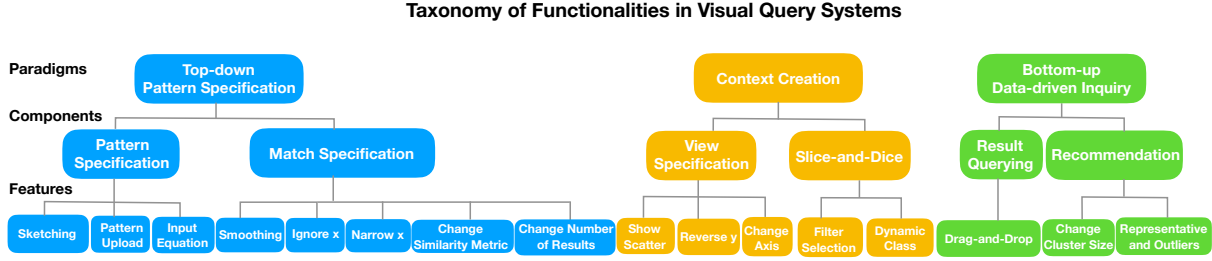


Figure 4: Taxonomy of functionalities in VQSs. From top, each of the three paradigm is broken down into key components in the system, which is instantiated as features in *zenvisage*. The bottom-most layer connects the use cases features that have practical or envisioned usage based on the evaluation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

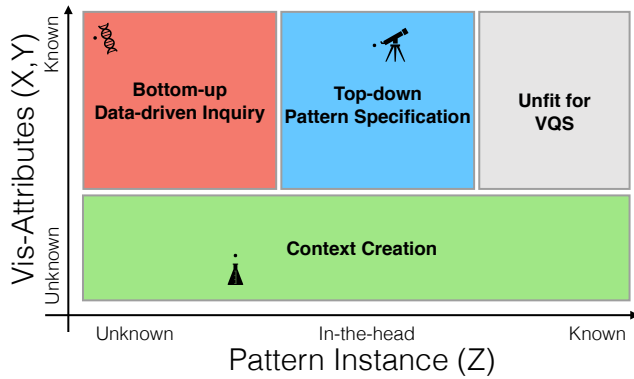


Figure 3: The design space of VQSs is characterized by how much the analyst knows about the visualized attributes and pattern instance. Colored area highlights the three different paradigms of VQSs. While prior work has focused solely on use cases in the blue region, we envision opportunities for VQSs beyond this to a larger space of use cases coverage in the red and green regions.

paradigm(*what, where, which*), its characteristic use case, and design challenges in supporting these paradigms.

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 an abstract conceptions based on external knowledge. The goal of top-down pattern specification is to address the *which* questions in visual sensemaking (*which pattern instance exhibits this pattern?*), effectively moving rightwards to the gray area in Figure 3, where the pattern instance is known. 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 the transient pattern 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 sense-making process enables users to go 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 goal of data-driven inquiry is to move towards the blue area in Figure 3 to help analysts gain more information about patterns of interest in-the-head. 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 analyst to navigate across different parts of the visualization collection. The goal is to learn about *where the patterns of interest lies*, effectively moving upwards in Figure 3 towards the known attributes region. 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 the context is dynamically reflected across other functionalities in the VQSs.

5 EVALUATION STUDY FINDINGS

Based on the audio, video screen captures, and click-stream logs recorded during the evaluation study, we performed thematic analysis through open-coding and categorized every

event in the user study as either a feature usage or an coded event label. Event codes included 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 categorized the features based on whether there was a sensible usage of the feature, as shown in Figure 8. A feature usage is marked as ‘Envisioned’ if the feature could be used practically 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 transcription to circumvent self-reporting bias, which can often artificially inflate the usefulness of the feature or tool under examination.

The evaluation study exemplified how participants adopted a diverse set of workflows based on the unique set of research questions they brought to the study. For example, in Figure 5, we find that astronomers largely focused on filtering, whereas material scientists made heavy use of dynamic class and geneticists made use of recommendations heavily. Participants intermixed functionalities across different sensemaking paradigms to address their problem contexts.

For the remaining paper, we will make use of results from thematic analysis to understand how feature usage informs the roles of each sensemaking paradigms in real-world analytic tasks.

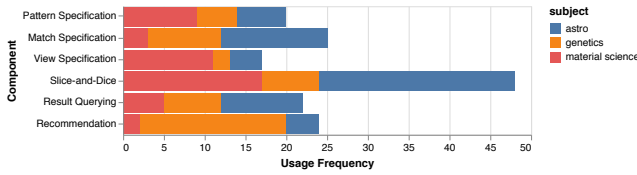



Figure 5: The number of times each component is used during the evaluation study, broken down by subject areas.

The Inefficiency of Sketching

To our surprise, despite the prevalence of sketch-to-query systems in literature, Figure 8 shows that only two out of our nine users had a practical usage for querying by sketching. The main reason why participants did not find sketching useful was that they often do not start their analysis with a pattern in mind. Later, their intuition about what to query is derived from other visualizations that they see in the VQS, in which case it made more sense to query using those visualizations as examples directly. In addition, even if a user has a query pattern in mind, sketch queries can be ambiguous or even impossible to draw by sketching (e.g. A2 looked for a highly-varying signal enveloped by a sinusoidal pattern indicating planetary rotation ).

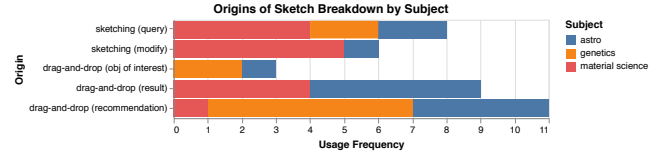


Figure 6: The number of times each sketch is generated by one of the workflows.

Given these initial findings, we further investigated where the ‘sketch’ (pattern on the canvas) originates from. In particular, Figure 6 shows that pattern queries can originate from either drag-and-drop or sketching. Within these two types of 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, described in the next section.

We note that there are 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 that was close enough to his desired visualization and then smoothed out the noise due to outlier datapoints by tracing a sketch over the visualization, 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, A3 was interested in pulsating stars that looked similar to stellar hotspots in terms of its dramatic amplitude fluctuations, but differ in that their patterns exhibited irregularities. Figure 7 (right) showed how she first picked out a regular pattern (suspected star spot), then modified it slightly so that the pattern looks more irregular.

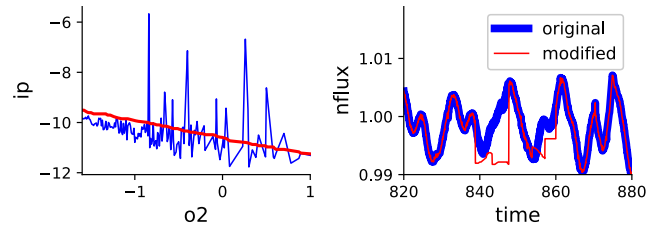


Figure 7: Canvas traces from M2 (left) and A3 (right) during the study demonstrating query modification. The original drag-and-dropped query is shown in blue and the sketch-modified queries in red.

The lack of practical use of top-down pattern specification is also reflected in the fact that none of the users queried by equation. In both astronomy and genetics use cases, 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.

Both query by sketch and equations adopt a problematic assumption that analysts start with a known and easy-to-specify search pattern in mind. 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 VQSs, since Table 1 show that past work have primarily focused on optimizing the components in the top-down paradigm, without regards to how useful these features are in real-world analytic tasks. We suspect that these limitation may be why existing sketch-to-query systems are not commonly adopted in practice.

Practical Use of Bottom-up approaches

Bottom-up data-driven inquiries are more common than top-down pattern specification when the users have no desired patterns in mind, which is commonly the case for exploratory data analysis. This is highlighted by Figure 8 which shows that top-down querying was only useful 29% of the use cases, where as it was useful for 70% of the use cases for bottom-up querying.

The prevalence of bottom-up approaches not only point to the need for result querying, but also providing recommendations for users without desired patterns in mind. As shown in Figure 6, the most common use of result querying comes from querying via a visualization that lie in a cluster. 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[10]. The clusters provoked G2 to generate a hypothesis regarding the properties of transients: “*Is that because all the transient groups get clustered together, 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. After browsing through a series query results and checking with an external database, he concluded that all stars have been incorrectly labelled with CLASS_STAR=0 as 1 during data cleaning. These examples show that both the browsing-act through recommendations and performing search via these results are essential for ‘closing the loop’ between the sensemaking acts in VQSs.

Likewise, many participants envisioned use cases for pattern loading. The ability to load in data patterns as a query would enable users to compare visualizations between different experiments, species, or surveys, query with known patterns from an external reference catalog, or verify the results of a downstream analysis. 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 [22], so querying with an exact pattern template would be helpful for distinguishing the patterns of interest from noise.

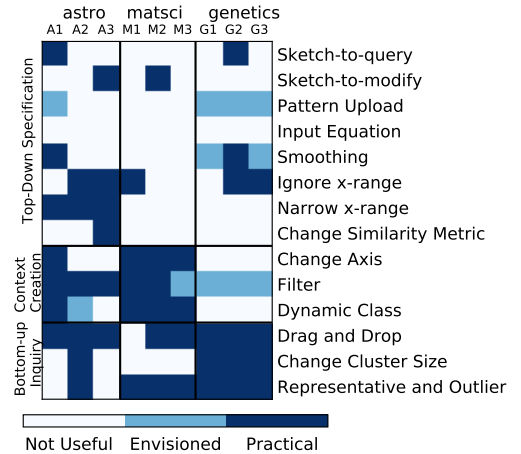


Figure 8: Heatmap of features categorized as practical usage, envisioned usage, and not useful.

Enriching Search with Context

Past studies in taxonomies of visualization tasks have shown that it is important to design features that enable users to select relevant subsets of data in visual analytics [2, 11]. Figure 8 shows that all participants either envisioned a use case or

utilized features in the context creation paradigm to explore and compare subsets of their data.

A1 expressed that even though the filtering step could be easily done with an external tool and reloaded into *zenvisage*, filtering on-the-fly was a powerful way to dynamically test his hypothesis. Interactive filtering lowers the barrier between the iterative hypothesize-then-compare cycle during sensemaking, thereby enabling participants to test conditions and tune values that they would not have otherwise modified. 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. 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 between 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 learned 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!

Context creation 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.

REFERENCES

- [1] 2016. Zillow. 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] Cecilia R Aragon, Sarah S Poon, Gregory S Aldering, Rollin C Thomas, and Robert Quimby. 2008. Using visual analytics to maintain situation awareness in astrophysics. In *Visual Analytics Science and Technology, 2008. VAST'08. IEEE Symposium on*. IEEE, 27–34. <https://doi.org/10.1088/1742-6596/125/1/012091>
- [4] Austin Nothhaft et al. 2015. Rethinking Data-Intensive Science Using Scalable Analytics Systems. *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data* (2015), 631–646. <https://doi.org/10.1145/2723372.2742787>
- [5] Jason Chuang, Daniel Ramage, Christopher Manning, and Jeffrey Heer. 2012. Interpretation and trust: Designing model-driven visualizations for text analysis. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 443–452. <https://doi.org/10.1145/2207676.2207738>
- [6] Ciolfi et al. 2016. Articulating Co-Design in Museums: Reflections on Two Participatory Processes. *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing - CSCW '16* (2016), 13–25. <https://doi.org/10.1145/2818048.2819967>
- [7] 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>
- [8] Yuri Demchenko, Paola Grosso, and Peter Membrey. 2013. Addressing Big Data Issues in Scientific Data Infrastructure. *Collaboration Technologies and Systems (CTS), 2013 International Conference on* (pp. 48-55). IEEE. (2013), 48–55. <https://doi.org/10.1109/CTS.2013.6567203>
- [9] Philipp Eichmann and Emanuel Zraggen. 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>
- [10] Brian S. Gloss, Bethany Signal, Seth W. Cheetham, Franziska Gruhl, Dominik C. Kaczorowski, Andrew C. Perkins, and Marcel E. Dinger. 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>
- [11] 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>
- [12] Harry Hochheiser and Ben Shneiderman. 2001. Interactive Exploration of Time Series Data. In *Discovery Science*, Klaus P. Jantke and Ayumi Shinohara (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 441–446.
- [13] 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.
- [14] 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>
- [15] M Kersten, S Idreos, S Manegold, and E Liarou. 2011. The Researcher's Guide to the Data Deluge: Querying a Scientific Database in Just a Few Seconds. *Proceedings of the VLDB Endowment*, (2011), 1474. <https://doi.org/10.1145/1409360.1409380>
- [16] 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>
- [17] 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>

- [18] Tamara Munzner Michael Sedlmair, Miriah Meyer. 2012. Design Study Methodology: Reflections from the Trenches and the Stacks. 1, 12 (2012), 2431–2440.
- [19] Mohebbi et al. 2011. Google correlate whitepaper. (2011).
- [20] Ivelina Momcheva and Erik Tollerud. 2015. Software use in astronomy: an informal survey. *arXiv preprint arXiv:1507.03989* (2015).
- [21] Michael J. Muller and Sarah Kuhn. 1993. Participatory Design. *Commun. ACM* 36, 6 (June 1993), 24–28. **<https://doi.org/10.1145/153571.255960>**
- [22] 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>**
- [23] 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.
- [24] 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>**
- [25] Prabhu et al. 2011. A Survey of the Practice of Computational Science. In *State of the Practice Reports (SC '11)*. ACM, New York, NY, USA, Article 19, 12 pages. **<https://doi.org/10.1145/2063348.2063374>**
- [26] 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>**
- [27] 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>**
- [28] Tarique Siddiqui, John Lee, Albert Kim, Edward Xue, Chaoran Wang, Yuxuan Zou, Lijin Guo, Changfeng Liu, Xiaofu Yu, Karrie Karahalios, and Aditya Parameswaran. 2017. Fast-Forwarding to Desired Visualizations with zenvisage. (2017). **<https://doi.org/10.1145/1235>**
- [29] Siddiqui et al. 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>**
- [30] 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>**