Understanding sketch-based visual query systems in action: A Case Study with Zenvisage

ABSTRACT

The increasing availability of rich and complex data in a variety of scientific domains poses a pressing need for tools to enable scientists to rapidly make sense of and gather insights from data. One proposed solution is to design visual query systems (VQSs) that allow scientists to interactively search for desired patterns in their datasets. While many existing VQSs promise to accelerate exploratory data analysis by facilitating this search, they are not widely used in practice. Through a year-long collaboration with scientists in three distinct domains—astronomy, genetics, and material science—we study the impact of various features within VQSs that can aid rapid visual data analysis, and how VQSs fit into scientists' analysis workflow. Our findings offer design guidelines for improving the usability and adoption of next-generation VQSs, paving the way for VQSs to be applied to a variety of scientific domains.

KEYWORDS

Visual analytics, visualization, exploratory data analysis, visual query, scientific data.
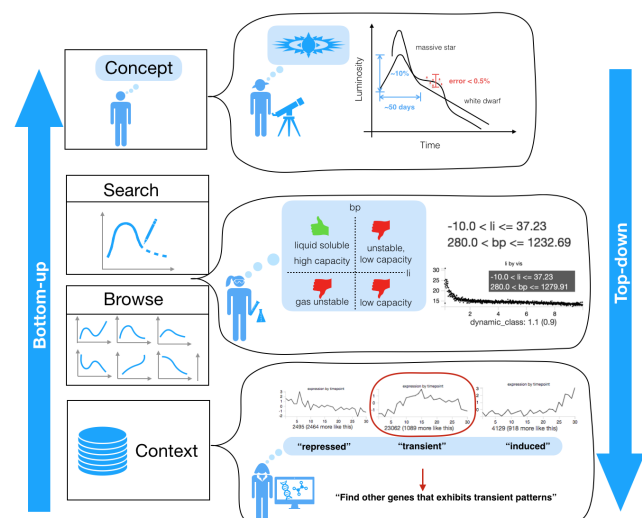
1 INTRODUCTION

Figure 1: Search Browse Model

2 RELATED WORKS

Visual query systems enable users to directly search for visualizations matching certain patterns through an intuitive specification interface. In the influential work on TimeSearcher [**?**], the query is composed of one or more box constraints. Similarly, in QuerySketch [31] and Google Correlate [19], the query is sketched as a pattern. Subsequent work have focussed on improving the expressiveness and flexibility of sketched queries through finer specification interface and pattern-matching algorithms, including specification of soft constraints [27] and implicit relaxed selection techniques [13]. Recent work have also performed crowdsourced perceptual studies to how humans rank similarity in patterns subjectively to develop novel specification interface and algorithms that are evaluated against existing 'flexible' metrics, such as Dynamic Time Warping (DTW) [6, 8, 16]. While these systems have shown to be effective for visual querying in controlled lab studies, they have not been evaluated in-situ on real-world use cases. In this work, we identify key components of VQSs beyond the sketch specification focus taken by existing work and develop an end-to-end process model for how analysts interact with VQSs in practice.

3 METHODS

Motivation

Visualization systems are often evaluated using controlled studies that measure the user's performance against an existing visualization baseline [26]. Techniques such as artificially inserting "insights" or setting predefined tasks for example datasets work well for objective tasks, such as debugging data errors [14, 23], but these contrived methods are unsuitable for trying to learn about the types of real-world queries users may want to pose on VQSs. Due to the unrealistic nature of controlled studies, many have proposed using a more multi-faceted, ethnographic approach to understand how analysts perform visual data analysis and reasoning [15, 17, 21, 26, 28]. In order to make the user study more realistic, we opted for a qualitative evaluation where we allowed participants to bring datasets that they have vested interests in to address unanswered research questions. Participatory design has been successfully used in the development of interactive visualization systems in the past [3, 4]. Sedlmair et al. [17] advocate that design study methodology is suitable for use cases in which the data is available for prototyping,

Table 1:

	Freehand Sketching	Shape Approx.	Range Selection	Flexible Matching	Filter Selection	Group Comparison	Concept Querying	Result Querying	Recommend Result
Timesearcher [11, 12]	Red	Green	Green	Red	Red	Red	Red	Green	Red
QuerySketch [31]	Green	Red	Green	Red	Red	Red	Red	Red	Red
QueryLines [27]	Green	Green	Green	Red	Red	Red	Red	Red	Red
SoftSelect [13]	Green	Green	Red	Green	Red	Red	Red	Red	Red
Google Correlate [19]	Green	Red	Red	Green	Red	Red	Green	Red	Red
TimeSketch [8]	Green	Red	Red	Green	Red	Red	Red	Red	Red
SketchQuery [6]	Green	Green	Red	Red	Red	Red	Red	Green	Red
Qetch [16]	Green	Green	Green	Green	Red	Red	Red	Red	Red
Zenvisage	Green	Green	Green	Green	Green	Green	Green	Green	Green

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, yellow represents processes: top-down querying, search with context, and bottom-up querying respectively, described more in Section 6.

but the task is only partially known and the information is partially in the user's head. In that regard, our scientific use cases with VQS is well-suited for a design study methodology, as we learn about the scientist's data and analysis requirements and design interactions that helps users translate their "in-the-head" specifications into actionable visual queries.

Participatory Design

We adopted a mixed methods research methodology that draws inspiration from ethnographic methods, iterative and participatory design, and controlled studies [18, 20, 28] to understand how VQSs can be used for scientific data analysis. Working with researchers from three different scientific research groups, we identified the needs and challenges of scientific data analysis and the potential opportunities for VQSs, 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. We initially spoke to analysts from 12 different potential application areas and narrowed down to three use cases in astronomy, genetics, and material science for our participatory design study. 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.

Given our early conversations with participants, we built a basic VQS to serve as the functional prototype in the design study. This early VQS prototype 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 [29, 30], which focused on the system and scalability aspects of the VQSs.

The use of functional prototypes is common and effective in participatory design to provide a starting point for the participants, as studied by Ciolfi et al.[5]. Our motivation for providing a functional prototype at the beginning of the participatory design sessions is to showcase capabilities of VQSs. Especially since VQSs are not common in the existing workflows of these scientists, participants may not be able to imagine their use cases without a starting point.

During 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 detailed timeline of our engagement with the participants and the features inspired by their use cases can be found in Figure 2. 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 the VQS prototype over the period of a year.

Evaluation Study

Finally, we conducted a realistic, qualitative evaluation 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
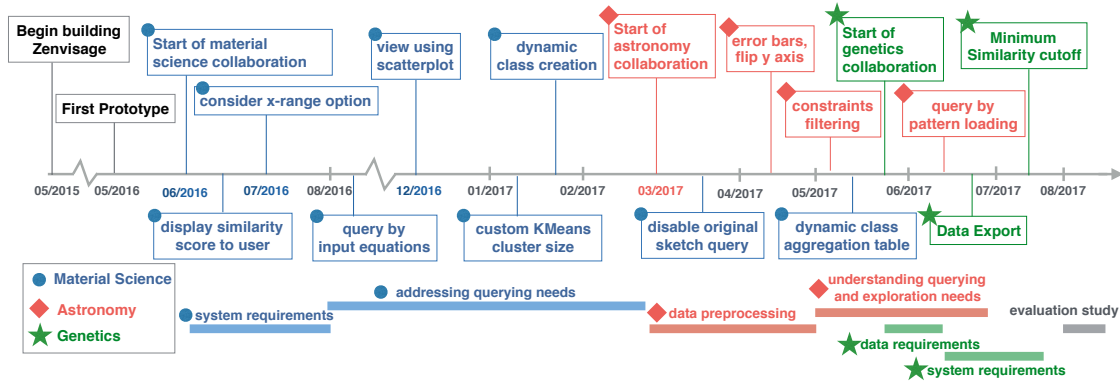
Figure 2: Participatory design timeline for the scientific use cases.

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 during the data exploration phase.

During the exploration phase, participants were informed that they could use other tools as needed. If the participant was out of ideas, we suggested one of the ten main functionalities in *zenvisage* that they had not yet covered. 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.

4 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 that participants have adopted for conducting certain analysis tasks.

Astronomy: The Dark Energy Survey (DES) is a multi-institutional project that surveys 300 million galaxies over 525 nights to study dark energy[7]. 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. The output is a time series of brightness observations associated with each object extracted from the images observed. For over five months, we worked closely with an astronomer on the project's data management team working at a supercomputing facility (A1). 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.

A1 was interested in *zenvisage* as he recognized how specific pattern queries could help scientists directly search for these rare objects. While an experienced astronomer who has examined many transient light curves can often distinguish an interesting transient object from noise by sight, they must visually examine and iterate through large numbers of visualizations of candidate objects. Manual searching is time-consuming and error prone as the large majority of the objects are not astronomical transients.

Genetics: Gene expression is a common data type used in genomics and is obtained via microarray experiments. The data used in the participatory design sessions was the gene expression data over time for mouse stem cells aggregated over multiple 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 [9, 24]. They were interested in using *zenvisage* to cluster gene expression data before conducting analysis with a downstream machine learning workflow.

To analyze the data, participant G1 loads the preprocessed data into a desktop application for visualizing and clustering gene expression data. Then she sets several clustering and visualization parameters before executing the clustering algorithm, then overlaid time series for each cluster is displayed

on the interface. G1 visually inspects that all 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 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 into a csv file 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, so that valuable meeting time was not wasted trying to regenerate results. The team had a vested interest in participating in the design of *zenvisage* as they saw how the interactive nature of VQSs and the ability to 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 datasets containing over 25 chemical properties for more than 280,000 different solvents obtained from simulations. We worked closely with a 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, and consists of 25 different chemical attributes. They wanted to use *zenvisage* to identify solvents that not only have similar properties to known solvents but also are more favorable (e.g. cheaper or safer to manufacture), as well as to understand how changes in certain chemical attributes affects them.

Participant M1 starts his data exploration process by iteratively applying filters on 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. Participant M1 was interested in using a VQS as it was impossible for him to uncover hidden relationships (such as how changing one attribute affects another attribute) between large number of solvents manually. Doris: Need to emphasize how known/unknown pattern and attribute for each usage case

5 STUDY FINDINGS

Themes Emerging from Participatory Design

We employed participatory design with our scientists to incorporate key features missing in our original VQS, and unaddressed in their existing workflows.

Exact Shape Specification interfaces allows users to submit a query through an exact description of a pattern, then the VQS returns a list of most similar matches. Almost all VQS supports freehand sketching on a virtual canvas through mouse or pen as a intuitive mechanism for specifying a desired patterns (Figure 3a). In addition to sketching, *zenvisage* also allows users to specify a functional form (e.g. $y=x^2$) for a pattern (Figure 3b). This feature was inspired by material scientists who were interested in finding solvents with known analytical models that characterize the relationships between their chemical properties.

Approximate Shape Specification While exact shape specification is an intuitive mechanism for constructing a visual query, as pointed out by past works [6, 13] pattern queries can be extremely imprecise. Many interfaces have developed constrained sketching mechanism to allow users to partially specify certain characteristics, such as angular slope queries for specifying the slope of a trend line [12] or piecewise trend querylines over a specified data range [27]. Both Qetch and *zenvisage* supports 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). We develop an interface for users to interactively adjust data smoothing algorithm and parameters on-the-fly to update the resulting visualizations accordingly (Figure 3c).

During participatory design, both material science and astronomy participants noted the difficulty of shape matching on their dense and noisy observational data and the challenge of picking the appropriate smoothing parameters during offline preprocessing. We found that tight integration between smoothing and visual search additionally tradeoff between the smoothness of the curve and the degree of approximation for shape-matching in VQSs. An over-smoothed visualization would return shape matches that only loosely resemble the query pattern. However, without smoothing, the noise may dominate the overall trend, which could lead to bad pattern matches.

Range Selection Often in time series analysis there are specific ranges of time and measure values with special domain specific significance that users may be interested in. One common specification on top of the exact or approximate shape query is limiting the pattern query matched to specific x or y ranges. Range selection can be performed through explicit textbox specification [16, 31], drawing line limits [27], or brushing interactions [11]. *zenvisage* employs the brushing mechanism to select desirable x-ranges to perform shape
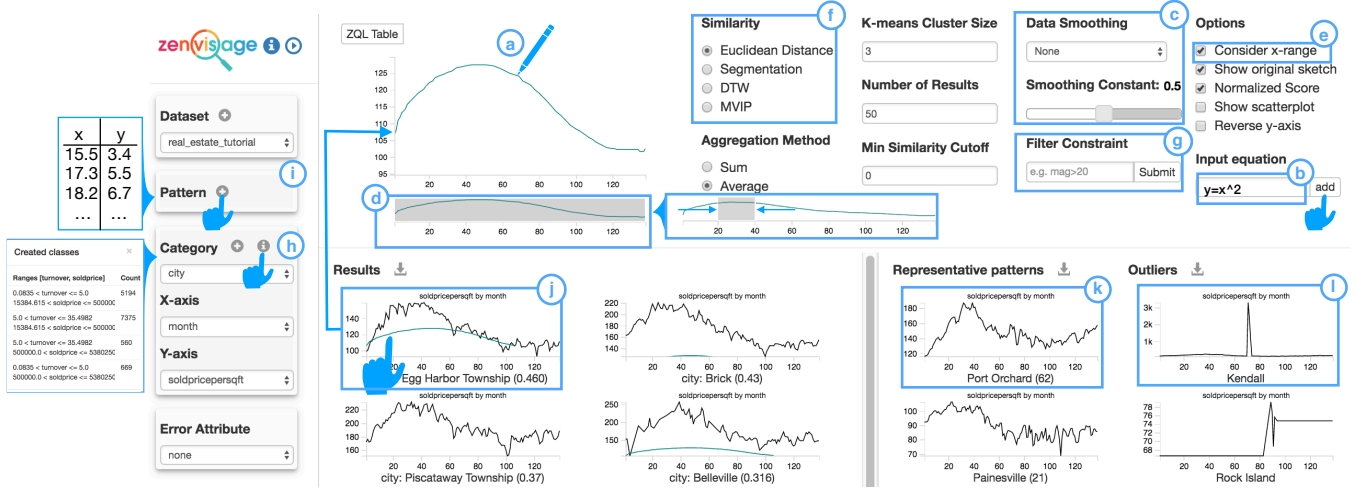
Figure 3: Our VQS after participatory design, which includes: the ability to query via (a) a sketch,(b) input equations, (i) drag and drop, or (j) uploaded patterns; (c) preprocessing via data smoothing; query specification mechanisms including (d) x-range selection and filtering, (e) x-range invariance, (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.

matching (Figure 3d). Additionally, y axis range selection could be performed through entering a filter constraint on the measure variable.

We chose to support only brushing for x, since it was more common to focus the context based on the independent variable in our use cases, such as zooming into particular sharp dips when looking for planetary transits or anomalous peaks indicative of erroneous experimental measurements. In contrast, y-range selection tends to be more global and enforced across multiple interaction sequences, such as looking for only signals above a certain threshold. The TimeSearcher and Queryline approach is most flexible for range selection as they allow composition of multiple range selection to formulate complex piecewise queries, such as finding a gene expression profile rising from x=1∼5 and declining from x=5∼10.

Flexible Matching Studies have shown that to facilitate subjectively meaningful pattern matches, VQSs need to support mechanisms for clarifying sketch interpretation and flexible shape matching algorithms [6, 8, 16]. In *zenvisage*, users has the option to change similarity metrics that perform flexible matching (Figure 3e), as well as the option to ignore the x-range in shape matching (Figure 3f). For finding supernovae, A1 primarily cared about the existence of a peak above a certain amplitude with an appropriate width of the curve, rather than the exact time that the event occurred, leading them to use the consider x-range feature. Geneticist G1 also expressed that she care about when the "trigger point" as long as the profile is "rising". This latter features is akin to the temporal invariants in SketchQuery [6].

Filter Selection 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,

10]. We find that users with large datasets first used their 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. To filter data, users could compose one or more conditions as filter constraints in a text field (Figure 3g). The filtering can be done on data columns associated with each pattern that is not visualized or on the visualized attributes. This feature is unique to *zenvisage* as most existing VQSs do not allow users to interact with data in the non-visualized columns.

Group Comparison A common analytical question from our participatory design is that users often want to bucket data points into customized classes based on existing properties, and subsequently compare between the customized classes. For example, material scientist 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 could browse 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 on-the-fly, effectively slicing-and-dicing the data based on their needs (Figure 3h). The information regarding the created classes is displayed in the dynamic class information table and as a tooltip over the aggregated visualizations.

Concept querying While input equations are useful when simple analytical models exist, this may not be true for other domains. In these scenarios, users can upload a query pattern of a sequence of points (Figure 3i). This is useful for patterns generated from advanced computational models used for understanding scientific processes or prelabelled data from an external reference database. For example, astronomer A1

can upload a query pattern based on synthetic light curves generated from simulations [22] or known supernovae that have been discovered in the past. Similarly, Google Correlate allows users to upload their own time series or enter search keywords that corresponds to a time series.

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 allows users instantiate timebox queries by dragging a result visualization and dropping into the query space; QuerySketch does so similarly through double clicking on the visualization. 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 3j). The distinction between concept querying and result querying is that concept querying loads in data that are external to the dataset from a reference source, whereas result querying initiates the query using visualizations created from the queried data source.

Recommendation displays visualizations that may be of interest to the users based on the context of the data. The recommendation feature is unique to *zenvisage*, which provides visualizations of representative trends based on clustering and highlights outlier instances that looks different from the rest of the visualizations (Figure 3k,l).

Evaluation Study Results

We recorded audio, video screen captures, and click-stream logs of the participant's actions during the evaluation study. We analyzed transcriptions of these recordings through open-coding and categorized every event in the user study as either a feature usage or via the coding labels:

- Insight (Science) **[IS]**: Insight that connected back to the science (e.g. "This cluster resembles a repressed gene.")
- Insight (Data) **[ID]**: Data-related insights (e.g. "A bug in my data cleaning code generated this peak artifact.")
- Provoke (Science) **[PS]**: Interactions or observations made while using the VQS that provoked a scientific hypothesis to be generated.
- Provoke (Data) **[PD]**: Interactions or observations made while using the VQS that provoked further data actions to continue the investigation.
- Confusion **[C]**: Participants were confused during this part of the analysis.
- Want **[W]**: Additional features that participant wants, which is not currently available on the system.
- External Tools **[E]**: The use of external tools outside of *zenvisage* to complement the analysis process.

We map the features to components based on the taxonomy described in Figure 8 to obtain Figure 4, which shows the timeline of event codes and component usage for each participant.

As shown in Figure **??**, to characterize the usefulness of each feature, we also categorized the features into one of the three usage types based on how each feature was used during the study:

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

We chose to derive 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. For the remaining paper, we will focus on understanding the design space of VQSs and highlight the takeaways of our study.
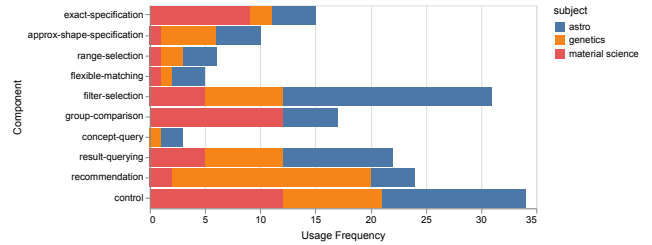

Figure 6: A really Awesome Image

6 DISCUSSION

Characterizing Design Space for VQSs

Visual querying often consists of searching for a desired visualization instance across a visualization collection (Z) for a fixed visualization axes (X,Y). To characterize the design space of VQS, we introduce two axes depicting the amount of information known about visualized attribute and pattern instance, as shown in Figure 7.

Along the **pattern instance** axis, the visualization instance may already be `known` to the analyst, exist as a pattern `in-the-head` of the analyst, or completely `unknown` to the analyst. For example, if a user wants to study only the pattern related to a specific gene, then the use cases is more suited for a visualization-at-a-time system, since the analyst can directly work with the visualization instance without the need for performing visual querying (Figure 7 grey). Most past VQSs assume that the analyst has a desired pattern in-the-head that could be conveyed through visual specification, such as a sketch. Inspired by Pirolli and Card's information foraging framework [25], which distinguishes between information processing tasks that are *top-down* (from theory to data) and
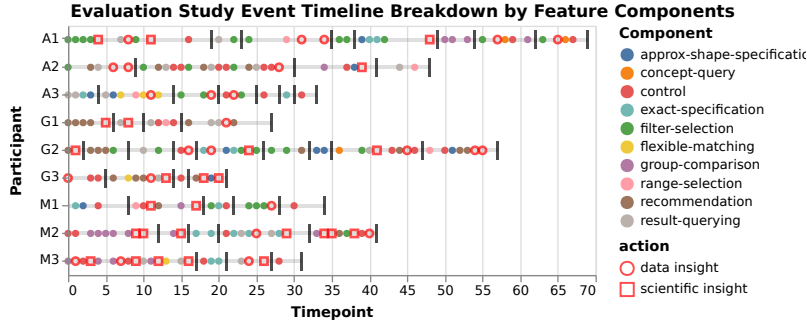
Figure 4: Timeline of event code and component usage. Black vertical tick indicates a session break, signaling the beginning of a new line of inquiry.
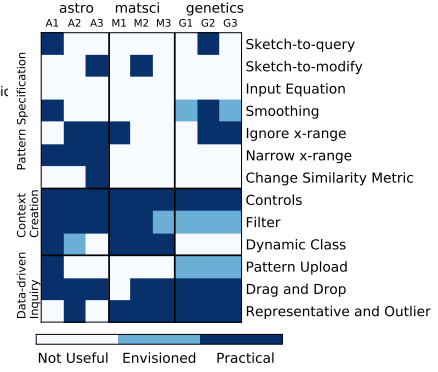

Figure 5: Heatmap of features categorized as practical usage (P), envisioned usage (E), and not useful (N).
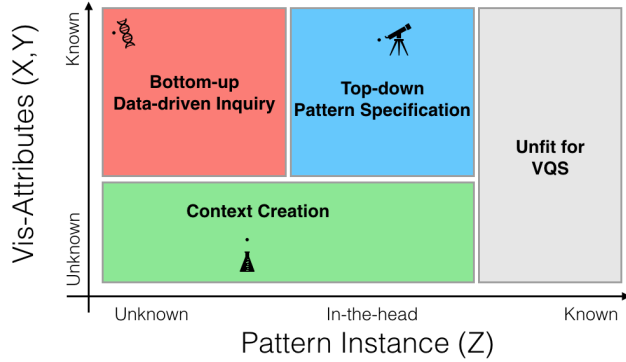

Figure 7: The design space of VQSs is characterized by how much the analyst knows about the visualized attributes and pattern instance. Colored areas highlights the three different paradigms of VQSs.

bottom-up (from data to theory), we define this process as "top-down pattern specification" (Figure 7 blue). On the other hand, in the realm of "bottom-up data-driven inquiry" (Figure 7 red), analysts often do not start with a known pattern instance. 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 VQSs.

The second axis **visualized attributes** depicts whether the analyst knows what X and Y axes she is interested in visualizing. Both the astronomy and genetics use cases, as well as past work in this space, had data in the form of a time series with known visualized attributes. In the case of our material science participants, they wanted to explore relationship between different X and Y variables. In the realm of unknown attributes, context creation (Figure 7 green) is essential for

allowing users to pivot across different visualization dimensions. While prior work has narrowed the focus of VQSs for use cases solely in the blue region, we envision opportunities for VQSs beyond this to a larger space of use cases and problems covered also by the red and green regions.
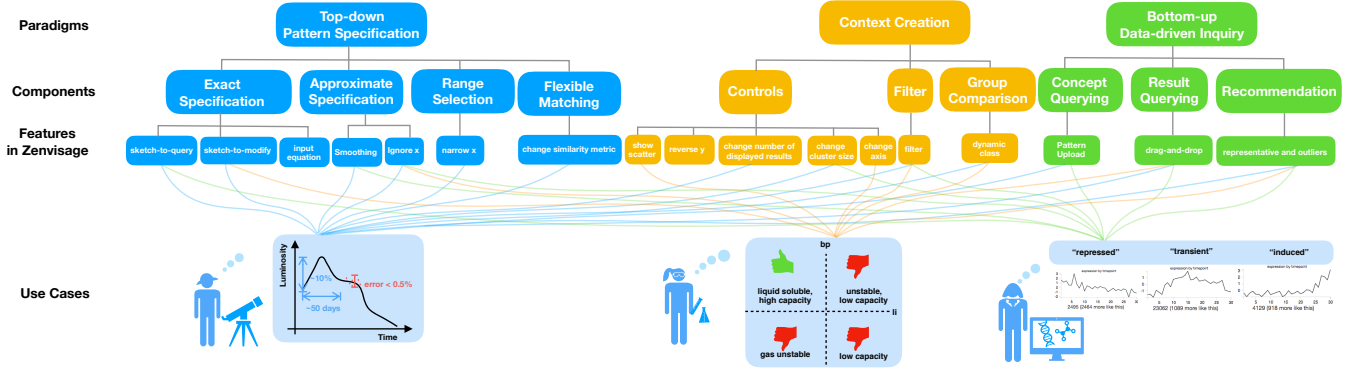
Design Challenges

Drawing from our participatory design experience, evaluation study, and literature review in this space, we design a taxonomy for understanding the essential functionalities in VQSs. In particular, we show how each use cases makes use of the different features in *zenvisage*. We organize the features into key components for VQSs as discussed in Section 5, which falls under the three paradigms in the VQS design space. The three paradigm that we have described are not mutually exclusive categories. In fact, we find that participants often construct a central workflow around one of the main paradigms and interleave variations with the feature usage from the two other paradigms as they iterate on the analytic task. The central paradigm adopted by each use case is tightly coupled with characteristics of the analytic challenges presented by the use cases. In this section, we will describe the design challenge (DC) involved in building features that support these paradigms by studying how participants interact with the VQS during the evaluation study.

Top-down Pattern Specification - typical astro use case - how to specify - given known pattern, what is the amplitude, noise tolerance, etc. of a sketch? How to translate the in-the-head query to visual query and how matching is done

involves tasks that narrow down the space of analysis

Bottom-up data-driven inquiry involves gathering more information during the analysis For example, genetics participants do not have a preconceived knowledge of what they want to search for in the dataset. They were mostly interested in gaining an overall sense what profiles exist in the

Figure 8: 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.

dataset through representative trends and therefore queried mainly through these recommended results to jumpstart further queries. The variations to their main workflow include changing cluster sizes and display settings to offer them different perspectives on the dataset and filtering on data attributes.

Context Creation involves spending time inspecting the results in more detail, including interpreting each visualization in greater detail or making plotting changes that offer another perspective (smoothing, display, and interpretability settings). pivoting changing across different axes,

visualizations to understand the relationships between physical variables. Participants are able to identify interesting relationships between physical variables when they examine each closely, but they are not sure what patterns to look for to begin with.

DP1: The Inefficiency of Sketch

In the context of visual querying, top-down approaches are query specification methods based on users' preconceived notion of what to search for. As shown in Figure **??**, this includes both components for specifying the pattern as well as controls governing the underlying algorithm of how the shape-matching is performed. Our interactions with the scientists showed that different modalities for inputting a query can be useful for different problem contexts.

To our surprise, despite the prevalence of sketch-to-query systems in literature, only two out of our nine users had a practical usage for querying by sketching. Overall, bottom-up querying via drag-and-drop was more intuitive and more commonly used than top-down querying methods, such as sketching or input equations.

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 (via drag-and-drop). In addition, even if a user has a query pattern in mind, sketch queries can be ambiguous or impossible to draw by sketching (e.g. A2 looked for a highly-varying signal enveloped by a sinusoidal pattern indicating planetary rotation).

Likewise, despite functional fitting being common in scientific data analysis, Figure **??** shows that querying by equation is also unpopular for similar reasons. In addition, the visualizations for both astronomy and genetics exhibit complex processes that could not be written down as an equation analytically. However, even when analytical relationships do exist (in the case of material science), it is challenging to formulate functional forms in an prescriptive, ad-hoc manner.

The latter case is also evident from the unexpected use cases where sketching was simply used as a mechanism to modify dragged-and-dropped queries. As shown in Figure 9 (top), M2 first sketched a pattern to find solvent classes with anticorrelated properties. However, the sketched query did not return the visualization he was interested in. So, he instead dragged and dropped one of the peripheral visualizations that was close enough to his desired visualization to the sketchpad and then smoothed out the noise due to outlier datapoints by tracing a sketch over the visualization. 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 characterized by dramatic changes in the amplitudes of the light curves. During the search, hotspots on stellar surfaces often show up as false positives as they also result in dramatic amplitude fluctuations, but happen at a regular intervals. In the VQS, A3 looked for

patterns that exhibits amplitude variations, but also some irregularities. As shown in Figure 9 (bottom), she first picked out a regular pattern (suspected star spot), then modified it slightly so that the pattern looks more irregular.
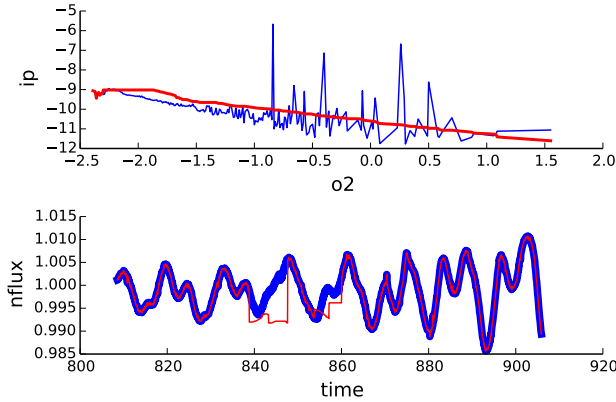
Figure 9: Examples of query modification by M2 (top) and A3 (bottom) performed during the study The inital drag-and-dropped query is shown in blue and the sketch-modified queries in red.

Both of these use cases 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*. Table 1 also show that most past work focus on optimizing the components in the top-down paradigm, missing out largely on the key components in the other two paradigms, indicated by the absence of green features on the right hand side of the table. We suspect that the limited coverage in addressing different types of analytics use cases may be why existing sketch-to-query systems are not commonly adopted in practice.

DP2: Usefulness of bottom-up approaches

While the usage of each querying feature may vary from one participant to the next, generally, result querying and pattern upload are considered bottom-up approaches that go from data to theory by enabling users to query via examples of known visualizations. For top-down approaches such as query by sketching and input equations, the user starts with an intuition about how their desired patterns should look like based on theory, then queries based on that. Our results indicate that *bottom-up querying approaches are preferred over top-down when the users have no desired patterns in mind*, which is commonly the case for exploratory data analysis.

Examples of practical uses of result querying includes inspecting the top-most similar visualizations that lie in a cluster and finding visualizations that are similar to an object of interest that exhibits a desired pattern. 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 (e.g. important genes of interest, objects labeled as supernovae), or verify the results of a simulation or downstream analysis by finding similar patterns in their existing dataset. In addition, users can also specify a more precise query that captures the essential shape features of a desired pattern (e.g. amplitude, width of peak), that cannot be precisely drawn as a sketched pattern. For example, Figure **??** (left) shows that the width of the light curve is characteristic to the type of supernovae that it is associated with, so querying with an exact pattern template would be helpful for distinguishing the patterns of interest from noise.

The prevalence of bottom-up approaches not only point to the need for supporting result querying in VQSs, but it also points to the need for providing recommendation for users who may not have a desired pattern in mind. We found that geneticists often gain their intuition about the data from the recommended representative trends. One example of rapid insight discovery 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[9].

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 attempted to identify different clusters. He proceeded by inspecting the visualizations in the cluster via drag-and-drop and 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, representative, and outlier 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 a filter constraint of CLASS_STAR=1. After a series of query results browsing and consultation with an external database, he concluded that the dataset had been incorrectly labelled with all the stars 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 loop in VQSs.

DP3: 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, 10]. We found that all participants either envisioned a use case or utilized components of the context creation paradigm offered in *zenvisage* to explore and compare subsets of their data.

A1 expressed that even though the filtering step could be easily done programmatically on the dataset 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, thereby enabling participants to test conditions and tune values that they would not have otherwise modified as much. 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?* `DIFFEXP=1` (G2) or *Can I find more supernovae candidates if I query only on objects that are bright and classified as a star?* `flux>10 AND CLASS_STAR=1` (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." As shown in Figures **??** and **??**, participants with the top-most provoked data actions (A1, A3, G2) made heavy use of filtering to explore different subsets of the data.

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. [...] And look how quickly we can do it! Quite good!

7 CONCLUSION

REFERENCES

[1] 2016. Zillow. `https://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] 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`

[5] 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`

[6] 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`

[7] Drlica Wagner et al. 2017. Dark Energy Survey Year 1 Results: Photometric Data Set for Cosmology. (2017). arXiv:1708.01531

[8] 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`

[9] 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`

[10] 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`

[11] 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.

[12] 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.

[13] 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`

[14] Sean Kandel, Andreas Paepcke, Joseph Hellerstein, and Jeffrey Heer. 2011. Wrangler: Interactive visual specification of data transformation scripts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 3363–3372. **https://doi.org/10.1145/1978942.1979444**

[15] 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**

[16] 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**

[17] Tamara Munzner Michael Sedlmair, Miriah Meyer. 2012. Design Study Methodolgy: Reflections from the Trenches and the Stacks. 1, 12 (2012), 2431–2440.

[18] Delbert C. Miller, Neil J. Salkind, and Delbert C. Miller. 2002. *Handbook of research design and social measurement*. SAGE.

[19] Mohebbi et al. 2011. Google correlate whitepaper. (2011).

[20] Michael J. Muller and Sarah Kuhn. 1993. Participatory Design. *Commun. ACM* 36, 6 (June 1993), 24–28. **https://doi.org/10.1145/153571.255960**

[21] 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**

[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] Patel et al. 2010. Gestalt: integrated support for implementation and analysis in machine learning. In *Proceedings of the 23nd annual ACM symposium on User Interface Software and Technology*. ACM, 37–46. **https://doi.org/10.1145/1866029.1866038**

[24] Pei Chen Peng and Saurabh Sinha. 2016. Quantitative modeling of gene expression using DNA shape features of binding sites. *Nucleic Acids Research* 44, 13 (2016), e120. **https://doi.org/10.1093/nar/gkw446**

[25] 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.

[26] 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**

[27] 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**

[28] 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**

[29] Tarique Siddiqui, John Lee, Albert Kim, Edward Xue, Chaoran Wang, Yuxuan Zou, Lijin Guo, Changfeng Liu, Xiaofo Yu, Karrie Karahalios, and Aditya Parameswaran. 2017. Fast-Forwarding to Desired Visualizations with zenvisage. (2017). **https://doi.org/10.1145/1235**

[30] 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**

[31] 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**