What Do We Mean When We Say "Insight"? A Formal Synthesis of Existing Theory

Leilani Battle and Alvitta Ottley

Abstract—Researchers have derived many theoretical models for specifying users' insights as they interact with a visualization system. These representations are essential for understanding the insight discovery process, such as when inferring user interaction patterns that lead to insight or assessing the rigor of reported insights. However, theoretical models can be difficult to apply to existing tools and user studies, often due to discrepancies in how insight and its constituent parts are defined. This paper calls attention to the consistent structures that recur across the visualization literature and describes how they connect multiple theoretical representations of insight. We synthesize a unified formalism for insights using these structures, enabling a wider audience of researchers and developers to adopt the corresponding models. Through a series of theoretical case studies, we use our formalism to compare and contrast existing theories, revealing interesting research challenges in reasoning about a user's domain knowledge and leveraging synergistic approaches in data mining and data management research.

Index Terms—Data and knowledge visualization, Visualization techniques and methodologies

1 Introduction

Ben Shneiderman famously said, "the purpose of visualization is *insight*, not pictures," underscoring a long-held conviction in the visualization community that the goal of the visualization tool is to enhance the user's understanding of the underlying data [1]. To this end, numerous scholars have aimed to understand the insight discovery process, yielding a wide range of *theories* defining an insight's structure, properties, and efficacy. These theories are critical in advancing insight-based research, producing models, processes, and metrics that researchers can adopt to derive meaning from their empirical observations. For example, researchers have used theory to infer a user's objective through recurring patterns in her logged interactions [2], [3], [4], [5], or to understand which of the user's utterances correspond to high-quality insights [6], [7].

However, the distinguishing characteristics of individual theories, the relationships between them, and their appropriateness for a given evaluation scenario are still unclear. For example, what does each theory mean by saying "insight"? When theories disagree, what are the benefits and drawbacks of adopting one over another? Finally, what gaps exist in the literature that new theories could fill?

In pursuit of these questions, we present a *theory-driven exploration* of existing work. First, we summarize existing arguments on the role and structure of insight. Then, we synthesize the core building blocks of insight based on observed overlaps in existing theories and contribute a formal specification for each building block. Finally, we compare the benefits and costs of several established theories of insight through case studies based on how they align with or deviate from our theorized building blocks.

We observe that researchers often differ in their terminology for theoretical concepts [8], [4], [9], making rote interpretation of the original papers a challenging strategy for making meaningful comparisons. Inspired by research in visualization specification [10], [11], we adopt a specification-based approach, where we use language-based specifications of the core building blocks of insight (as synthesized from the literature) to implement the structures implied in the original papers. Then, we analyze the resulting structures to identify interesting agreements and/or disagreements between existing definitions. Through these specifications, we place each theory on equal ground and make precise theoretical comparisons. Furthermore, we publish each implemented case study as a self-contained, executable program that can be evaluated or reused by the visualization community in the future[1].

Based on our findings, we highlight several exciting research opportunities, such as formally defining and capturing domain knowledge, sharing insight data alongside system interaction logs, and broader use of innovations in data mining and data management across the visualization community. We discuss these challenges further in section 6.

In summary, this paper makes four contributions:

- We present an in-depth **review of existing insight definitions and connect them** with related concepts in insight discovery, such as analysis *tasks*.
- Based on observed agreements in the literature, we **synthesize the core building blocks of insights** and contribute formal specifications of these building blocks.
- We introduce **four use cases for applying our formalism** to a range of insight-based theories and studies.
- We **discuss open challenges in insight- and theory-based visualization research** motivated by this work.

2 How are Insights Currently Defined?

In this section, we review the visualization literature to understand how insights are conceptualized and to identify *agreements* and *disagreements* among existing theories. Note that this work summarizes a prior literature review [12].

Leilani Battle is with the University of Washington, Seattle (e-mail: leibatt@uw.edu). Alvitta Ottley is with Washington University in St. Louis (e-mail: alvitta@wustl.edu).

1. https://osf.io/t9e63/?view_only=9857ef52c3334739b0001dd6d7cf324c

2.1 Review Process

We conducted keyword searches for "visualization insight" and "visualization task" in Google Scholar; we reviewed the proceedings of VIS and EuroVis from 2013 to 2023; and we noted relevant papers with "insight" in the title or abstract. This generated an initial list of 125 papers. Then, we filtered for only papers that explicitly investigate how to define, analyze, or discover insights, e.g., "But what, exactly, is insight? How can it be measured and evaluated?" [13]. For each relevant paper, we reviewed its list of references to identify papers we may have missed. To explore potential relationships between insights and existing task theory, we also considered papers cited by our initial list that explicitly defined visualization objectives, tasks, or reasoning models. These steps yielded a list of 38 papers. With feedback from colleagues/reviewers, we extended it to include their suggestions, producing a final list of 42 papers. The two authors collaboratively analyzed how insight was defined in each paper, focusing on high-level themes as well as key characteristics of insights. We cite synergistic ideas when relevant, e.g., knowledge graphs and visualization recommendations.

2.2 Categories of Insight

The prior work details several high-level categories of insights, distinguishing *instantaneous sparks* ("aha" moments) and long-term *knowledge building*. For example, Chang et al. [14] distinguish between a "knowledge-building insight," or information that extends a user's existing knowledge structures, and "spontaneous insight," or a "eureka" moment that reorganizes loosely related knowledge. We focus on knowledge-building insights in this paper, but our formalism could be extended to spontaneous insights.

We also observe categorizations focused on the *source* of insights, such as the input dataset, social structures of the analyst, or an analyst's domain knowledge. For example, Saraiya et al. define four categories of data-driven insights [15], [16]: overall distributions, patterns, grouping, and detail. Choe et al. [17] extend these ideas by providing more granular categorizations, such as distinguishing distributions versus data summaries or correlations versus general trends. Zgraggen et al. [7] follow a similar structure but focus on categorizing the ways people make data comparisons to extract data-driven insights. Note that these categories are not mutually exclusive and may co-occur [18].

However, data-driven insights are not the only insights an analyst may uncover. For example, Gotz et al. [19], Pousman et al. [20], Liu and Heer [21], Choe et al. [17], and Karer et al. [22] observe that analysts often connect what they see in the data with their own knowledge and experiences, i.e., with *domain knowledge* that exists outside the target dataset. Pousman et al. broaden this view to support insights that may not be purely data-driven, in particular "awareness insight," "social insight," and "reflective insight" [20].

The many variations in how insights are categorized suggest that insights may not be an atomic unit in themselves. Instead, it may be more useful to categorize the *components* of insights, such as the types of knowledge gleaned from data-driven or domain-driven sources [22]. We take this into consideration in our formalism and subsequent analysis.

2.3 The Varying Definitions of Insight

We also observe inconsistent *definitions* for what constitutes an insight. Are they utterances, statistical correlations, or something more complex? In this section, we summarize the range of definitions proposed in the literature to identify key components relevant to developing a unified formalism.

Some definitions assert that **insights are utterances**. For example, in the context of a user study, Saraiya et al. define insight as "an individual observation about the data by the participant, a unit of discovery," [15], [16] which can include "any data observation that the user mentions" during lab studies [15], [21], [7], [23] and self-reported insight diaries from field studies [24] and competition submissions [25].

Insights are often categorized by how their calculation supports users' hypotheses, claims, and reflections, pointing to a second definition – **insights are data facts**. For exmaple, Choe et al. propose eight insight classes, where six classes are statistical in nature ("trend," "correlation," "data summary," "distribution," "outlier" and "comparison") and two are adapted from existing taxonomies ("detail" [15], [24] and "self-reflection" [20]). Zgraggen et al. propose five insight classes, all of which are statistical in nature [7]: "shape," "mean," "variance," "correlation," and "ranking". This consistent grouping suggests that statistical representations, i.e., *data facts* may be a core building block of insights. Chen et al. formalize the relationship between data facts and insights through their Fact Management Framework [26], which provides a theoretical base for defining insights.

The prior work also suggests that **insights are hypotheses and/or evidence.** For example, Sacha et al. argue that users leverage analysis findings primarily as evidence to support, refute, or generate new hypotheses [27]. To evaluate how study participants perform during open-ended exploration tasks, Gomez et al. label each observed insight from their study as a "claim," i.e., "a general hypothesis, question, or remark about the data model that is potentially synthesized from multiple observations," or as "evidence," such as an observation comprised of "specific references to data points" that support the claim [28]. Guo et al. [29] and Liu and Heer [21] adopt a similar evidence- and hypothesis-based framing for insights, respectively. Thus, this definition seems to build on the concept of data facts.

Finally, the prior work also suggests that **insights are knowledge links.** In particular, Chang et al. argue that visual analytics research "considers insight more or less as units of knowledge" [14]. Others refine this idea further by defining insights as *links* that connect analysis *findings*, such as visualizations and statistical results [30], [31], with user *knowledge* (e.g., [15], [6], [19], [32], [33], [27], [34], [22]), such as knowledge synthesized from the current session or earlier sessions (e.g., [35], [14], [36], [18], [37]), or a priori knowledge the user brings to the exploration process (e.g., [24], [19], [32], [20], [33], [38], [22]).

These links can be implicit, such as when observed through qualitative studies [15], [24], [18], or explicit, such as when users apply annotation interactions [19], [36], [39] or link interactions [35], [19], [36], [34], [37], [40] to connect system visualization state with concepts recorded in their own digitized notes. Furthermore, insights can be hierarchical and build on one another over time, increasing the

complexity of subsequent insights [15], [24], [6], [20], [36], [18], [34], [41], [38].

Moreover, Pike et al. argue for more formal semantics for capturing user insights, which can enable visual analysis systems to more effectively process, reason about and even extract new insights [42]. Moreover, Smuc et al. argue that insights are better analyzed by explicitly tracking how they build on one another and propose relational insight organizers (or RIOs) to organize and visualize the resulting insight graph [18]. RIOs share similarities with the structures proposed by Gotz et al. [19], where user knowledge is also captured as a graph, with high-level concepts and instantiations of these concepts representing nodes in the graph, and links between instances and/or data representing edges in the graph. Similar graph-based structures have also been suggested by Shrinivasan and van Wijk [35], Willett et al. [39], Mathisen et al. [41], and He et al. [37], as well as in the intelligence analysis literature [43].

Integrating the Definitions: These definitions may appear distinct. However, a close look at the varying perspectives points to an overarching theme – **an insight is a collection of knowledge**. Although existing definitions vary in what they emphasize, we find that the components themselves appear to be consistent across definitions, which we categorize as *analytic* and *domain knowledge*. For example, analytic knowledge consistently includes data facts, generalizations, and hypotheses. Domain knowledge includes domain expertise and personal experiences. Awareness of these components enables us to navigate these varied definitions of insight. Further, our proposed formalism unites these perspectives by distinguishing between these knowledge sources and defining concrete links between them.

2.4 Scoping Insights

Insight discovery generally occurs within a certain visual analysis scope [36], [4] which is often tightly bound with the definition of tasks [4], [44] or objectives [9], [45] in visualization research. For example, many theories categorize the scope of insights that analysts may be looking for. These theories often take the form of task *taxonomies* and *typologies* [4], where specific tasks observed in the field or lab studies are abstracted into task classes, such as "Find Anomalies" [46], "Search/Comparison" [47] or "characterizing data distributions and relationships" [8]. Specific to insights, several taxonomies target common insight generation *processes* to predict insight scope, rigor, and complexity [32], [29], [8].

Task models may also take the form of *frameworks*, where the scope, structure, and relationships between of observed tasks, are abstracted into general-purpose hierarchies. Examples include the framework of tasks, sub-tasks, actions, and events proposed by Gotz and Zhou [36], and the goals to tasks framework proposed by Lam et al. [45]. We observe that these models predict the scope of insights by culling the set of relevant data facts (taxonomies) or narrowing the range of data for applying these data facts (frameworks). However, these models only represent a range of possibilities. They are inappropriate for describing the *exact* insights an analyst uncovers while completing a visual analysis task. We take these strengths and limitations into account by defining both a data scope and method scope within our formalism for insights in section 4.

An analyst's interest in pursuing certain tasks can also be defined with respect to the kinds of insights they *expect* to uncover. This observation stems from the idea that a user's data analysis strategy is likely informed by an initial goal or "hunch" regarding the target dataset [45], [7], [8], even if only vaguely at first [8]. For example, Bertin defines tasks according to the structure of the underlying data and the information the user seeks to learn from this data [48]. Andrienko and Andrienko extend Bertin's ideas to define tasks as declarative functions over data relations comprised of *targets*, i.e., data attributes of interest, and *constraints*, i.e., query predicates over these attributes [49]. We note that Andrienko and Andrienko and Bertin's proposals overlap significantly with established definitions of task in database research, notably *relational calculus*, a core component of the *relational model* that also defines tasks (or queries) as declarative functions over data relations [50]. That being said, existing declarative definitions of task are limited to scoping the user's *expectations* and fail to encapsulate the insights that the user actually found, which are often the focus of insight discovery work.

To summarize, existing task models are useful aids for inferring insight scope but are insufficient for fully defining insights. Thus, we focus our formalism on scoping insights directly rather than indirectly through task-based models.

3 Formalism Design Goals

This paper aims to synthesize existing theories into a unified formalism that represents the core building blocks of insight. Such a formalism could provide theoretical consistency and structure to future insight-based theories and evaluations. To guide our development of a new formalism, we summarize three recurring principles from the current theory that clarify the scope, structure, and complexity of insights. Alongside each principle, we propose a research goal that informs our formalism in section 4.

Principle 1: Insights Represent Linked Units of Knowledge. We observe in the literature that insights establish *links* between the user's data manipulations, data observations, and their knowledge of related phenomena, and new insights often link back to old ones as analysts' understanding of the data evolves (subsection 2.3). As a result, insights are generally complex objects with multiple components. An insight is not just a piece of data but the user's *interpretation* of this data, which may involve connections with prior knowledge. These components appear consistent across the literature but may vary in name/terminology (subsection 2.2).

It is challenging to capture these complex relationships within an atomic insight definition. Instead, we formalize the constituent parts of insight and their relationships. This idea resembles a well-known problem in visualization specification. Overlaps in visualization taxonomies suggest core components that can be formalized into guiding theory, e.g., the Grammar of Graphics [51], which in turn guide the development of visualization grammars such as Vega-Lite [10]. To this end, we establish our first research goal:

> **Research Goal 1 (RG1):** Rather than defining insights as atomic units, focus on the low-level components of insight and their relationships.

Principle 2: Insights Distinguish Domain Knowledge From Analytic Knowledge. The literature suggests that insight is not just the simple linking of two nodes in a graph. Insights occur at specific points within the analysis process, particularly when the user's prior experience intersects with the data at hand in a meaningful way. For this to occur during visual analysis, the visualizations must extend the user's knowledge base to bring in new information (subsection 2.2), such as by revealing new observations or facts about a dataset (subsection 2.3). We define *analytic knowledge* as statistical, visual, or factual information that can be derived directly from the target dataset. For example, we can observe whether crime is trending up or down in a given city by tallying reported crimes. Then, the user can connect these findings with their own knowledge and experiences to clarify the significance of analytical results. Inspired by the concept of domain knowledge in knowledge modeling research [52], we define *domain knowledge* as contextual information that cannot be inferred from the target dataset. For example, crime datasets may tell us *what* crimes were reported but not *why* they occurred; this additional context requires a deductive approach to analyzing the crime data. Insights can also build upon each other, where higher-level insights likely draw on existing links between analytic (inductive, bottom-up) and domain (deductive, top-down) knowledge (subsection 2.3). Based on these ideas, we propose our second research goal:

> **Research Goal 2 (RG2):** Insights link analytic and domain knowledge, requiring the ability to distinguish between them while preserving the relationships and patterns that bind them together.

Principle 3: Insight Discovery is About the Interpretation *of Visualizations, Not the Visualizations Themselves.* We observe that studying insight discovery requires more than just collecting the visualizations that users create. If visualization is about insight, not pictures [1], then insight discovery is about tracking how people *interpret* the pictures [53], [3], [22], not tracking the pictures themselves nor the corresponding interactions. In other words, recording visualization provenance, i.e., how visualizations are created over time, is a poor substitute for recording actual insights. To understand the relationship between visualizations and insights, we need to be able to track them in equally fine detail. We observe established methods [54] and formalisms [4] for visualization provenance but a dearth of counterparts for insights, leading to our final research goal:

> **Research Goal 3 (RG3):** Insights represent the *interpretation* of data, e.g., knowledge gleaned from a visualization, not just the visualization itself.

4 A Formalism for Visualization Insights

As a first step towards clarifying the value of existing insight theory, we formalize the theoretical building blocks of insight as agreed upon in the literature. In this section, we define the formal structure of each building block and identify the theories from which they were sourced.

4.1 Initial Building Blocks

Principle 1 highlights the prevailing assumption that insights connect knowledge. This assumption implies a notion of *links* between knowledge units. Although links are discussed frequently in the visualization literature (see subsection 2.3), they lack a formalism defining their structure. The closest we observe is proposed by Kandogan et al. [31], in which they suggest that knowledge links could be defined using data mining structures, specifically, *knowledge graphs*. Inspired by their approach, we contribute a knowledge graph-driven formalism of links, supporting **RG1**. We draw on existing research in knowledge graph construction from the data mining [55] and knowledge modeling communities [52]. We briefly introduce knowledge graphs from the data mining and knowledge modeling literature and defer to surveys for more details [55], [56], [52].

4.1.1 Defining Knowledge Graphs

A knowledge graph is "a graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent potentially different relations between these entities" [55][2]. Entities, the nodes of a knowledge graph, can take various forms, including physical objects such as cities and people [55], as well as digital objects such as web pages and structured datasets [57]. The edges of a knowledge graph are used to record relationships between entities [55], [56], [52], such as which cities are located within certain countries or hyperlinks between web pages (e.g., Wikipedia pages) and their backing data (e.g., Wikidata). In knowledge modeling research, knowledge nodes can also be designed around a framework, where nodes of the same type share the same descriptive attributes defined by the framework [52].

Visualization research in insight discovery frequently discusses how insights drawn from visualizations can extend a user's knowledge base (see subsection 2.3), which can be represented as establishing new *nodes* N and *links* (or edges E) within a knowledge graph $G(N, E)$. To capture this agreement, we treat analytic and domain knowledge as *knowledge nodes* $n \in N$ within our formalism. Knowledge nodes can connect to other nodes of the same type, for example, by specifying edges $e \in E$ between related domain knowledge nodes. We also define higher-level nodes that group domain and analytic knowledge nodes together, such as to specify a higher level *insight node*. Together, *knowledge nodes* and *knowledge links* represent our initial building blocks for defining insights.

4.1.2 Defining Knowledge Nodes and Links

Our formalism uses a graph-based data model to represent insights, where the nodes N in the graph represent units of (explicit [58]) knowledge, and the edges E represent the relationship between nodes, which may be directed or undirected. An example is shown in Figure 1, which is based on the Baltimore crime data exploration example by Mathisen et al. [41]. We see that a knowledge node can record historical or domain-specific knowledge about related concepts, such as majority black ("The majority of people living in Baltimore, MD are Black or African American"). Similarly, we can record knowledge gained by analyzing Baltimore

2. Note that "knowledge graph" and "knowledge base" can be used interchangeably here. We defer to this survey [56] for more details.

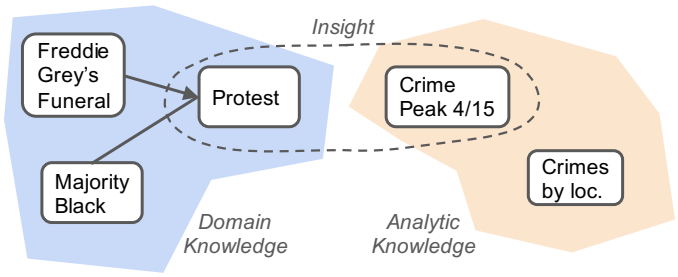


Fig. 1: Knowledge nodes can be linked to form insights, e.g., linking observations of peaks in total crimes with related characteristics and historical events for Baltimore. Undirected edges capture relatedness. Directed edges record trajectories of knowledge expansion in the graph.

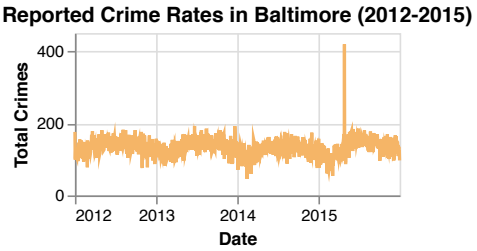


Fig. 2: A peak in reported crimes is observed in April 2015.

crime data as shown in Figure 2 (e.g., Crime Peak 04/15 "There was a spike in reported crimes in April 2015"). The edges in the graph represent relatedness and can track the evolution of knowledge acquisition. For example, an undirected edge can indicate the relatedness of Protest — Crime Peak 04/15 or a directed edge can indicate that the protests were in response to the unwarranted killing of Freddy Grey, an unarmed black man, by the Baltimore Police Freddy Grey's Funeral —> Protest. Similarly, if analyzing total crimes by location (on the street, in the home, etc.) turns out to be a "dead end" then this analysis may not extend the user's knowledge base, resulting in a node Crimes by loc. with no edges in the graph. Furthermore, rather than linking protests directly with crime peaks in the graph, we could group them into a higher-level graph node, such as an insight node, shown as a dashed line in Figure 1.

We define knowledge links as pairs of knowledge nodes:

$$\texttt{KnowledgeLink} := (n_i, n_j),\ n_i \in N, n_j \in N$$

For directed links, we can further specify `source` and `target` nodes as part of the knowledge link:

$$\begin{aligned}\texttt{DirectedKnowledgeLink} &:= (\texttt{source}, \texttt{target})\\ \texttt{source} &:= n_i \in N\\ \texttt{target} &:= n_j \in N\end{aligned}$$

To streamline our formalism, we combine knowledge nodes and links within a single definition for *knowledge nodes*:

TABLE 1: Example table `T` for creating an `instance` representing the 2015 Baltimore Protests using Equation 2.

Source	Link
Wikipedia	`https://en.wikipedia.org/wiki/2015_Baltimo...`
Vox	`https://www.vox.com/2016/7/27/18089352/fre...`

$$\begin{aligned}\texttt{KnowledgeNode} &:= \{name, \texttt{sources}, \texttt{targets}, \texttt{related}\}\\ \texttt{sources} &:= \{n_i,\ n_j,\ ...\} \subseteq N\\ \texttt{targets} &:= \{n_k,\ n_l,\ ...\} \subseteq N\\ \texttt{related} &:= \{n_p,\ n_q,\ ...\} \subseteq N\end{aligned} \tag{1}$$

where *name* is a string identifier for the specified knowledge node, and other nodes that contributed directly to the current node can be specified as a set of `sources`. Nodes influenced by the current node can be specified as a set of `targets`. We can specify nodes in an undirected relationship as a set of `related` nodes. In the remainder of the paper, we refer only to our definition of knowledge nodes, which implicitly contains our definition of knowledge links.

4.2 Formalizing Domain Knowledge

Revisiting Principle 2, we observe an emphasis in the literature on distinguishing different types of knowledge, such as meaningful statistical or mathematical features that can be derived from a target dataset, i.e., *analytic knowledge*, and contextual information (e.g., personal experience, domain expertise) that cannot be derived from this dataset, i.e., *domain knowledge*. In Figure 1, events such as Freddy Gray's funeral represent domain knowledge regarding the history of Baltimore (highlighted in blue). The statistics calculated using Baltimore crime reports represent analytic knowledge (highlighted in orange), e.g., that April 2015 contained some of the highest days of reported crimes between 2012 and 2015. In our formalism, we capture this distinction by formalizing the defining characteristics of domain knowledge and analytic knowledge nodes, supporting **RG2**.

We integrate existing ideas to formalize our collective understanding of domain knowledge. Gotz et al. [19] suggest that analysts reason about different *concepts* as they analyze a dataset, as well as specific *instances* of these concepts that stand out. Extending these ideas, we treat *concepts* as custom type definitions representing a belief or idea that the analyst wants to express, such as the concepts of "racism", "conspiracy", or "protest". We define an `instance` as a representative case of a *concept* and the supporting data `T`:

$$\begin{aligned}\texttt{instance} &:= \{name,\ concept,\ \texttt{T}\}\\ \texttt{T} &:= \{a_1,\ a_2,\ ...\}\end{aligned} \tag{2}$$

where `T` is a relational table with at least one data attribute/column (a_1, a_2, etc.) and row. Revisiting our running example, suppose we learn about the 2015 Baltimore Protests through Wikipedia and Vox. We can record this in `T` using two attributes (a_1 = `Source`, a_2 = `Link`) and two rows for the corresponding articles, shown in Table 1.

Using the concept of *frames* from the knowledge modeling literature [52], we formally define a `DomainKnowledgeNode` by extending our

KnowledgeNode definition from Equation 1 to include properties from our instance definition in Equation 2:

$$\texttt{DomainKnowledgeNode} := \{name,\ \texttt{sources},\ \texttt{targets},\ \texttt{related},\ concept,\ \texttt{T}\} \tag{3}$$

In this way, we can capture meaningful concepts and instances expressed within a unit of domain knowledge as well as track their propagation through a knowledge graph.

4.3 Formalizing Analytic Knowledge

Analytic knowledge lacks a precise definition in the literature. For example, Gotz et al. appear to define analytic knowledge as annotations to domain knowledge [19]. Alternative definitions proposed by Shrinivasan and van Wijk [35] and Andrienko and Andrienko [59] seem to suggest that analytic knowledge is information gained from querying and interacting with a dataset. These ideas also inspired our third research goal (**RG3**): to distinguish users' creation of visualizations from their interpretation of visualizations. In section 3, we define analytic knowledge as information derived through the manipulation of data such as through interacting with visualizations. Towards formalizing this definition, we first clarify what we mean by "information derived" (i.e., *data relationships*) from "manipulation of data" (i.e., *data transformations*).

4.3.1 Data Relationships

We deduce from the prior work that when researchers encounter analytic knowledge in insight-based studies (see subsection 2.3), they tend to interpret it in terms of *mathematical or statistical characteristics*, such as by recording associated data correlations, distributions, patterns, or anomalies observed by users [15], [32], [23], [26], [17]. We refer to these characteristics as ***data relationships***, but they are also referred to as data facts in the literature (see subsection 2.3). One could simply try to track the visualizations a user created as a proxy for data relationships. However, users can easily draw different conclusions from the same visualizations [60], making visualizations ambiguous records of analytic knowledge [53]. Our goal is not to collect pictures but to understand the knowledge gleaned from them. For these reasons, we define analytic knowledge in terms of *properties that can be calculated or modeled directly from the underlying dataset*. We stress that our formalism represents a *quantitative interpretation* of the analytic knowledge gained and not the raw utterances of study participants.

There are many ways in which researchers have interpreted data relationships, such as by creating statistical representations like histograms [2], linear regression models [61] or statistical sketches [62], [26] or even machine learning models like Hidden Markov models [3], [63] or support vector machines [2]. Amid this panoply of techniques, we observe a recurring high-level structure that we leverage to formally define the theoretical properties of data relationships.

Given a relational table T with data attributes $\texttt{A} = \{a_1, a_2, ..., a_n\}$, we find that ***multivariate data relationships***: (1) take one or more data attributes as input variables for training and one attribute as an output variable for prediction; (2) define a training function to build a model that predicts the output given the input; and (3) define a prediction function that uses the model to map new inputs to projected outputs:

$$\begin{aligned} \texttt{MultivariateRelationship} &:= \{f_{train},\ f_{predict}\} \\ f_{train}/f_{predict} &:= \texttt{A}_{ij} \mapsto a_o,\ \texttt{A}_{ij} \subset \texttt{A},\ a_o \in \texttt{A} \end{aligned} \tag{4}$$

The training function can be as simple as calculating coefficients for a linear equation, but it can also be complex, such as training machine learning models. We find that ***univariate relationships*** such as kernel density estimation: (1) take a single input attribute to simulate; (2) define a function to train a model to capture the corresponding distribution; and (3) define a function to simulate records from the modeled distribution:

$$\begin{aligned} \texttt{UnivariateRelationship} &:= \{f_{train}, f_{simulate}\} \\ f_{train} &:= a_i \mapsto \emptyset, a_i \in \texttt{A} \\ f_{simulate} &:= \emptyset \mapsto a_i \end{aligned} \tag{5}$$

This definition covers all of the data relationships observed in our literature review, and easily extends to new data relationships. For example, any new multivariate relationship that takes one or more attributes as input, predicts an attribute as output and provides the requisite function types are automatically covered under our formalism.

4.3.2 Data Transformations

Typically, analysts must process their data to facilitate insight discovery [64], [65], such as by filtering, aggregating, sorting, etc., which we refer to as ***data transformations***. For example, to answer the question *"which k dates have the most reported crimes?"* we have to group the Baltimore crime data by date, count all reported crimes per date, then sort by the count to retrieve the top k dates.

Data transformations can have profound effects on an analyst's ability to extract insights. For example, Battle and Heer found that differences in users' interaction sequences in Tableau could lead to different queries being executed over the data and ultimately different answers to the same analysis tasks [8]. Furthermore, interfaces that hinder interactive data processing have been shown to negatively impact insight generation [21], [23]. Given the critical role of data transformations in insight discovery, we consider them essential to formalizing analytic knowledge.

Similar to Andrienko and Andrienko [49], we specify data transformations as queries over relational tables. We use *relational algebra* to represent data transformations, where relational algebra can be considered the dual to *relational calculus* [50] (see subsection 2.4 for more on relational calculus). To do this, we treat data transformations as a sequence of relational algebra operations, where each operation is essentially a function that takes a relational table T as input and returns a relational table $\texttt{T}'$ as output:

$$\begin{aligned} \texttt{DataTransformation} &:= [o_1, o_2, ..., o_i, ...] \\ o_i &:= \texttt{T} \mapsto \texttt{T}' \end{aligned} \tag{6}$$

where T may not have the same attributes or rows as $\texttt{T}'$. For example, to calculate peak crime dates in Baltimore, we:

group by date, count reported crimes per date, and sort the dates by count. The aggregation produces fewer attributes and rows than the original input table since we are grouping all reported crimes by date and returning a single count per date. In contrast, the sort returns an output table with the same shape as its input, since the rows are simply reordered.

As hinted at by Bertin [48] and Andrienko and Andrienko [49], tracking data transformations can provide an advantage over interaction logs, since it emphasizes a user's interpretation of the data rather than the idiosyncrasies of a particular user interface. For example, Tableau desktop offers a myriad of ways to filter a dataset through its interface but they all map to the same filtering operations in relational algebra [64], [8]. Although our formalism differs from traditional SQL, it matches related languages such as Microsoft LINQ [66], which can be mapped to SQL [67].

4.3.3 Analytic Knowledge Nodes

Similar to domain knowledge nodes (subsection 4.2), we formalize analytic knowledge nodes by merging our base definition for knowledge nodes (Equation 1) with our definitions for data transformations (Equation 6) and data relationships (Equation 4 and Equation 5):

$$\begin{aligned} \texttt{AnalyticKnowledgeNode} &:= \{name,\ \texttt{sources},\ \texttt{targets},\\ &\quad\ \texttt{related},\\ &\quad\ \texttt{dataTransformation},\\ &\quad\ \texttt{dataRelationship}\}\\ \texttt{dataTransformation} &:= [o_1, o_2, ..., o_i, ...]\\ \texttt{dataRelationship} &:= \{f_{train},\ f_{predict}\}\\ &\quad\ |\ \{f_{train},\ f_{simulate}\} \end{aligned} \tag{7}$$

With this definition, researchers gain access to a precise, quantitative representation of a user's analytic knowledge. We record exactly how the data has been manipulated and can quantify the knowledge we believe the user gained from their results. Further, this representation remains consistent regardless of differences in user interfaces, enabling it to generalize across visualization tools.

4.4 Formalizing Insights

With precise definitions for domain (Equation 3) and analytic (Equation 7) knowledge, we complete our formalism for insights. As shown in Figure 1, we view insights as a *cluster* of relevant domain and analytic knowledge nodes within the user's knowledge graph. However, simply drawing a link between the target nodes renders this *hierarchical* relationship indistinguishable from other relationships in the graph. To express the hierarchical nature of insight, we define it as a *higher level representation* of knowledge:

$$\begin{aligned} \texttt{InsightNode} &:= \{name,\ \texttt{sources},\ \texttt{targets},\\ &\quad\ \texttt{related},\ \texttt{domainKnowledge},\\ &\quad\ \texttt{analyticKnowledge}\}\\ \texttt{domainKnowledge} &:= \{\texttt{domainKnowledgeNode}_1,\\ &\quad\ \texttt{domainKnowledgeNode}_2, ...\}\\ \texttt{analyticKnowledge} &:= \{\texttt{analyticKnowledgeNode}_1,\\ &\quad\ \texttt{analyticKnowledgeNode}_2, ...\} \end{aligned} \tag{8}$$

where more than one node can be specified in `domainKnowledge` and `analyticKnowledge` to support more complex insights. Insights can also be hierarchical by integrating other insights [68], [18], [19] (see Figure 3).

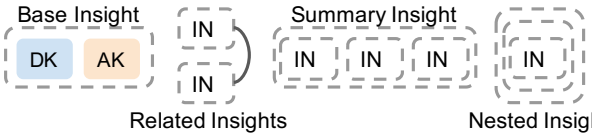


Fig. 3: Various insight structures enabled by our formalism. "DK" is domain knowledge, "AK" analytic knowledge, and "IN" insight.

To support these relationships, we define insights as higher level *graph nodes* that inherit from `KnowledgeNode` (Equation 1), supporting linking and extension of insights as the user's knowledge graph evolves over time (see Figure 3). Using our formalism, we can link any previous insights as `sources` that informed the current insight and new insights as `targets` that were informed by the current insight.

4.5 Implementation

To support transparency, evaluation, and reuse of our formalism by the research community, we have implemented a specification language called PYXIS that is freely available online. Pyxis can be used to specify *all* of the building blocks defined above: concepts, instances, domain knowledge, data transformations, data relationships, analytic knowledge, and insights. Pyxis is implemented in TypeScript and can be used in TypeScript and JavaScript projects. Pyxis supports both the Node and Observable JavaScript environments.

Rather than re-implementing known data transformations, we import existing libraries into Pyxis using wrapper classes matching the specifications in subsubsection 4.3.1 and subsubsection 4.3.2. Our codebase imports the Vega transforms [69] and Arquero [70] libraries, so all data transformations supported by Vega or Arquero can also be used in Pyxis. Likewise, we import a wide range of existing data relationships, such as linear regression models, anomaly detection through isolation forests, and univariate distributions via kernel density estimation through wrapper classes that integrate existing libraries.

5 PRACTICAL USE CASES

In this section, we present four use cases demonstrating how to apply Pyxis. We also highlight future research opportunities enabled by our formalism. The corresponding Pyxis code is shared in our supplemental materials.

5.1 Use Case 1: Recreating an Analysis Session

In this use case, we show how to use Pyxis by recreating our ongoing example from Mathisen et al. [41], which follows a fictional analyst, John, as he investigates crime peaks in Baltimore from 2012 through 2015. John's analysis uncovers the first crime peak on April 27, 2015 which coincides with protests sparked by the funeral of Freddie Gray, a young black man who was killed by the Baltimore police. An overview is provided in Figure 4, where each node represents a particular Pyxis object that we create throughout the example, and the edges between nodes represent relationships between objects.

5.1.1 Specifying Domain Knowledge

Pyxis enables us to formalize John's domain knowledge as abstract *concepts*, e.g., the "protests" and specific *instances* of these concepts, e.g., the "Baltimore protests." We define the concept "Protest" (lines 1-4). Then, we define an instance of the "Protest" concept based on the Baltimore Protests in 2015 using a new domain knowledge node (lines 5-21).

```
1 const protest = new Concept(
2   "Protest", // name
3   [] // parentConcepts
4 );
5 const protestsNode = new
  ↪  DomainKnowledgeNode(
6     "2015BaltimoreProtests", // name
7     protest, // associated concept
8     { // metadata
9         attributes: [{
10            name: "Source",
11            type: nominal
12        }, {
13            name: "Link",
14            type: nominal
15        }],
16        values: [{
17            "Source": "Wikipedia"
18            "Link": "https://en.wikip..."
19        }]
20    }
21 );
```

Since domain knowledge inherits from knowledge nodes (see subsection 4.2), we can also specify node relationships, such as one node "causing" or being "related to" another node. For example, to link a source (i.e., parent) node, we call the `addSource` function on the `protestsNode` object.

5.1.2 Specifying Analytic Knowledge

To develop analytic knowledge, analysts infer *data relationships* (subsubsection 4.3.1). To prepare the data for visualization or modeling, analysts apply *data transformations* (subsubsection 4.3.2). We demonstrate how to create data transformations by using Arquero to calculate total reported crimes per day and identify peak crime days.

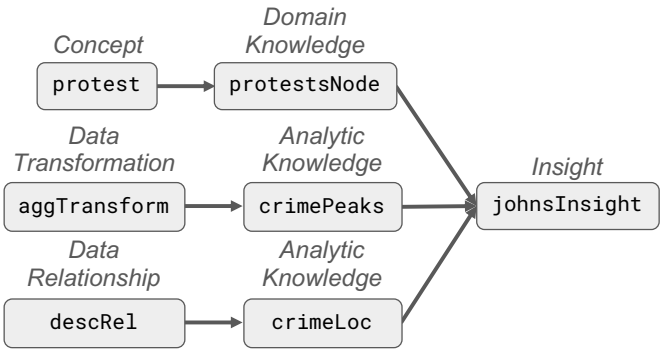


Fig. 4: An implementation of our running example from section 4. Each node represents a specified Pyxis object and its corresponding component from our formalism. The directed edges represent input relationships. For example, the `protest` object is an input to the `protestsNode` object.

```
1 // Arquero Data Transformation
2 const aggTransform = {
3     sources: [baltimoreCrime],
4     transforms: [
5       // group by day
6       { op: "groupby", args:
        ↪  ["CrimeDate"] },
7       // count crimes per day
8       { op: "rollup", args: [{ count:
        ↪  op.count() }] },
9       // sort days by count
10      { op: "orderby", args:
        ↪  [desc("count")] },
11      // return top 3 days with highest
        ↪  counts
12      { op: "filter", args: [() =>
        ↪  op.rank() <= 2] }
13    ]
14 };
15  const getAggTransformResults =
16   () => executeDataTransformation(
     ↪  aggTransform);
```

First, we specify the input dataset (line 3). Then, we group reported crimes by date (line 6), count total records per day (line 8), sort the dates by count (line 10), and filter for the top three dates with the highest counts (line 12). Finally, we execute the transformations using the `executeDataTransformation` method on line 16.

```
1 const crimePeaks = new
  ↪  AnalyticKnowledgeNode(
2   "peakCrimes", // node name
3   Date.now(), // timestamp
4   aggTransform, // data transformation
5   null, // data relationship
6   getAggTransformResults, // results
7 );
```

Next, we record how John processed the data to identify peaks in a new analytic knowledge node. First, we give this analytic knowledge a name (line 2) and record when John learned it (line 3). Then, we connect relevant data transformations and/or relationships (lines 4-5). In this case, John's findings relate only to the `aggTransform` object. Similar to domain knowledge nodes, we can link to other analytic knowledge nodes using `addSource`, `addTarget`, and `addRelated`. All node objects share this property.

To demonstrate data relationships in Pyxis consider this extension of the Baltimore example: suppose John is curious whether location is indicative of crime type, for example, whether different crimes happen indoors versus outdoors, or in an apartment versus a business. We can specify a new model to predict this relationship as follows:

```
1 const descRel = new
  ↪ DecisionTreeClassification(
2   "predictCrimeType", // name
3   [ // input attributes to predict with
4     {
5       name: "Inside/Outside",
6       attributeType:
          ↪ AttributeType.nominal
7     },
8     {
9       name: "Premise",
10      attributeType:
          ↪ AttributeType.nominal
11    }
12  ],
13  // output attribute to be predicted
14  {
15    name: "Description",
16    attributeType: AttributeType.nominal
17  }
18 );
19 dt.train(baltimoreCrimes.records);
20 const prediction = dt.predict(
  ↪ baltimoreCrimes.records[0]);
```

Since the input and output attributes are categorical, we specify a decision tree classifier to predict their relationship (line 1). The input attributes used to train the model are "Inside/Outside" and "Premise" (lines 3-12). The output attribute being predicted is "Description" (lines 14-17), which describes the type of crime reported. However, using a machine learning relationship is not required, and the model type can easily be swapped in Pyxis by choosing a relationship type other than `DecisionTreeClassification`. Line 19 shows how we can train the specified decision tree model on the Baltimore crimes dataset and Line 20 shows how this model can be used to predict the crime type of specific records. In this way, specified analytic knowledge can be evaluated for statistical rigor by testing the accuracy of the underlying data relationships, supporting prior calls for more precise evaluation of user insights [7].

Suppose that `Inside/Outside` and `Premise` are not strong predictors of crime `Description`. We can record this result in a new AnalyticKnowledgeNode as follows:

```
1 const crimeLoc = new
  ↪ AnalyticKnowledgeNode(
2   "crimeLocations", // node name
3   Date.now(), // timestamp
4   null, // data transformation
5   descRel, // data relationship
6   null // results
7 );
```

Pyxis supports any type of multivariate relationship, including K nearest neighbors, linear regression, and naive Bayes models, as well as univariate relationships (e.g., via kernel density estimation) and other statistical relationships such as outliers (e.g., via isolation forests).

5.1.3 Specifying Insight

The last step in specifying insights is to link domain knowledge with relevant analytic knowledge. Continuing our example, we specify an insight connecting John's domain knowledge about the Baltimore protests (line 4) and analytic knowledge regarding peak crime dates in Baltimore (line 5):

```
1 const johnsInsight = new InsightNode(
2   "johnsInsight", // name
3   // domain knowledge
4   [protestsNode],
5   // analytic knowledge
6   [crimePeaks, crimeLoc]
7 );
```

We can also keep track of John's "dead ends" as desired, for example including our `crimeLoc` analytic knowledge node as shown on line 4. Similar to the domain and analytic knowledge nodes, insight objects also support the linking of source, target, and related insights.

5.2 Use Case 2: Analyzing Insight Complexity

Similar to insight scope, insight complexity is an important concept in the literature. For example, insight complexity is used to estimate the quality or value of insights [6], [18], [37]. In this use case, we explore how North defines insight complexity [6] and connect this definition with alternative conceptualizations of insight complexity in the literature.

5.2.1 Three Levels of Complexity

North argues that "complexity is determined by how much data is involved in the insight" and gives three forms of "insights" over monthly rents as examples [6]. We implement these insights (see Figure 5) using data from the U.S. Department of Housing and Urban Development[3].

A) *"Simple" Insights.* North's simplest insight computes minimum and maximum rent values, which we calculate using an aggregate data transformation.

B) *"More Complex" Insights.* North proposes a "more complex" insight that estimates a normal distribution over the rent data, which we calculate as a univariate data relationship using Vega's statistics package[4].

C) *"Even More Complex" Insights.* The most complex insight estimates the shape of the rent distribution using a histogram. This maps to a series of data transformations to bin the data and then aggregate it by bin ranges.

5.2.2 Insight Complexity ⊆ Query Complexity?

According to our formalism, North's examples involve a univariate data relationship (a normal distribution) and two sets of data transformations (min/max calculations and binned aggregation). Further, North seems to emphasize the data transformations, which are essentially relational queries over data (see subsection 4.3). We see this in Figure 5, where binned aggregation requires a more complex sequence of operations compared to the min/max calculations. Here, "insight complexity" could mean *relational query complexity*, which aligns well with existing definitions of insight posed by Andrienko and Andrienko [49] and Bertin [48] as well as relevant database research [71] and even research on program complexity [72]. In this sense, the Andrienko and Andrienko [49] and Bertin [48] definitions

3. https://www.huduser.gov/portal/datasets/50per.html
4. https://github.com/vega/vega-statistics/

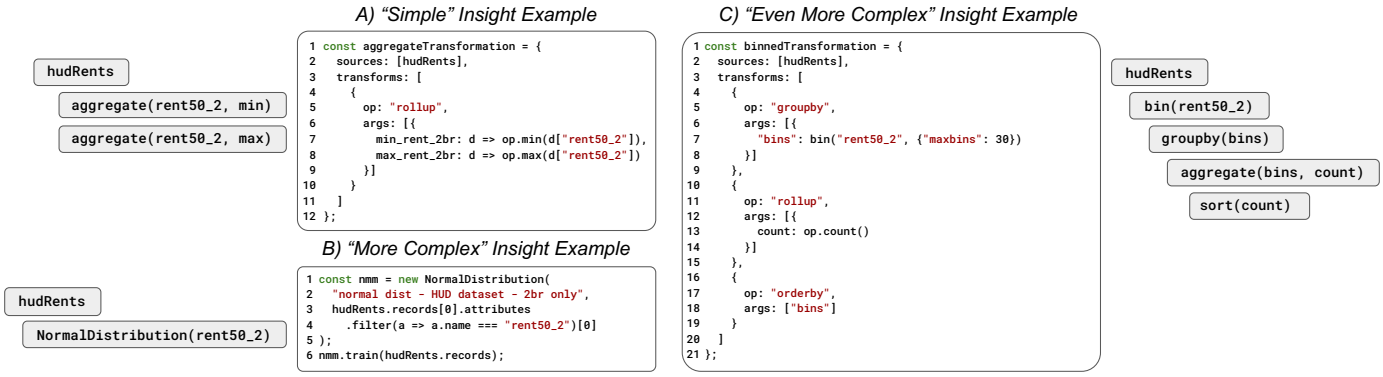


Fig. 5: North's three levels of insight complexity [6] in Pyxis alongside the corresponding relational operators.

could be considered generalizations of North's definition of insight complexity since relational queries can encompass a wider range of programs than min-max or histogram calculations. Queries also provide an easy means of calculating data coverage, which North uses to define complexity.

That being said, defining insight complexity as query complexity makes data manipulation the focus of insight rather than how people interpret data, i.e., as *data relationships*. Thus, this view of insight complexity is deficient, as hinted at by North's inclusion of a univariate data relationship. Saraiya et al. [15] and Smuc et al. [18] extend this idea to include multivariate relationships in their definitions. Kandogan and Engelke take this a step further by applying relational query patterns to express data relationships such as linear correlations [31], further enriching our understanding of insight complexity. With a richer definition of insights comes the need for alternative methods for measuring insight complexity, which could be an exciting opportunity for future research.

5.2.3 Benefits of the Formalism

Using our formalism, we could hypothesize new measures of insight complexity, such as by measuring knowledge *depth* by computing the longest path from a knowledge node to its earliest ancestor, which aligns with existing definitions of exploration depth [8]. Using Pyxis, knowledge node depth can be calculated programmatically by backtracking from a node's directed edges. This calculation can also be augmented to include the number of relational operations involved in connected data transformations. Consistent with North's assertion, we could even measure knowledge *breadth* as the percentage of data values (rows $\times$ attributes/columns) involved as inputs and outputs to the corresponding data transformations and/or relationships.

5.3 Use Case 3: Analyzing Participants' Insights

Although insight-based user studies have been critical to understanding how people form insights, participants' insights are generally self-reported, requiring a means of *validating* insight quality. Traditionally this has been done by hand [15], [24], [6], [21], [29]. However, recent considers how to partially automate the validation process [7], [8], [37].

Pyxis provides a convenient structure for validating participants' insights and could facilitate further automation. We demonstrate this benefit by recreating insights reported by Battle and Heer from their study of how analysts explore data in Tableau [8]. We focus on task "T3" for the wildlife strikes dataset[5], which asks: *"What relationships (if any) do you observe involving weather conditions and strike frequency, or counts over time?"* The task also specifies three attributes to consider: `precip`, `sky`, and `incident_date`. We recreate two *contradictory* answers observed by Battle and Heer: (A) strikes are not correlated with time (reported by 13 participants) and (B) bad weather leads to more strikes over time (reported by 3 participants). We use Pyxis to shed light on the discrepancy among participants (see Figure 6.)

A) *Analyze the `precip` Attribute.* First, we analyze the incidence of wildlife strikes using the `precip` attribute by applying a series of data transformations to: remove null `precip` entries on lines 4-7, extract the year from each `incident_date` on line 11, and count the total incidents observed per year, grouped by `precip` conditions (e.g., "fog," "rain," etc.) on lines 15-26. Overall, we see that incidents do *not* appear to increase with time, with the exception of "rain" conditions, shown in Figure 6b. We can record these findings in a new analytic knowledge node named `precipNode` in Figure 6a.

B) *Analyze the `sky` Attribute.* Second, we repeat this analysis, but replace `precip` with `sky` on lines 6 and 12 in our code, denoted in yellow in Figure 6a. In this case, we see a steady increase in incidents per year for all observed weather conditions, shown in Figure 6c. We can record these findings in a new analytic knowledge node named `skyNode` in Figure 6a.

5.3.1 Benefits of The Formalism

We see that even when performing the same data transformations, participants could still derive drastically different answers based on which attributes were analyzed. These results suggest that even if participants' analyzed both attributes while completing T3, their answers were likely influenced by which attribute they favored, `precip` or `sky`. Since Battle and Heer focused on analyzing interaction sequences [8], they may have overlooked potential *structural similarities* in participants' answers. By using a consistent structure to represent analytic knowledge, we see how the data can influence which conclusions participants draw regardless of which interactions were performed.

5. https://wildlife.faa.gov/search

A) First, We Analyze the Incidence of Strikes Using the **precip** *Attribute*

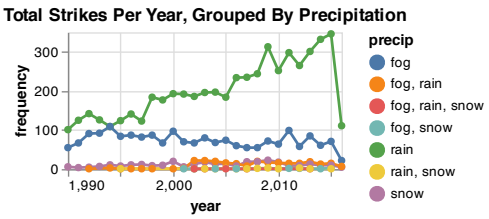
```
1 const precipTransformation = {
2   sources: [birdstrikes],
3   transforms: [
4     {
5       op: "filter",
6       args: [d => d.precip !== null &&
7         op.lower(d.precip) !== "none"]
8     },
9     {
10      op: "derive",
11      args: [{
12        year: d => op.year(d["incident_date"]),
13        precip: d => op.lower(d.precip)
14      }]
15    },
16    {
17      op: "groupby",
18      args: ["year","precip"]
19    },
20    {
21      op: "rollup",
22      args: [{ frequency: op.count() }]
23    },
24    {
25      op: "orderby",
26      args: ["year"]
27    }
28  ]
29 };
```

```
1 const precipNode = new AnalyticKnowledgeNode(
2   "precipKnowledge", // name
3   Date.now(), // timestamp
4   precipTransformation, // transformation
5   null, // data relationship
6   // [optional] description
7   "Incidents do not increase with time."
8 );
```

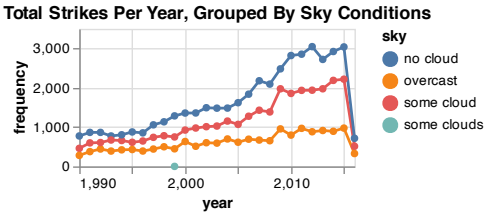
B) Second, We Analyze the **sky** *attribute and Compare.*

```
1 const skyNode = new AnalyticKnowledgeNode(
2   "skyKnowledge", // name
3   Date.now(), // timestamp
4   skyTransformation, // transformation
5   null, // data relationship
6   // [optional] description
7   "Incidents increase over time."
8 );
```

(a) Specified data transformations (left) and analytic knowledge nodes (right). Only lines 6-7 and 13 need to change to analyze the `precip` or `sky` attribute, shown in yellow.



(b) Analyzing the `precip` attribute. Strikes *do not* appear to increase with time.



(c) Analyzing the `sky` attribute. Strikes appear to *increase* with time.

Fig. 6: In a prior study [8], participants explored wildlife-aircraft strikes under various weather conditions, `precip` and `sky`. Holding data transformations constant **(a)**, we posit that participants who focused on the `precip` attribute **(b)** versus the `sky` attribute **(c)** likely drew different conclusions, showing how attribute selection can influence insight discovery.

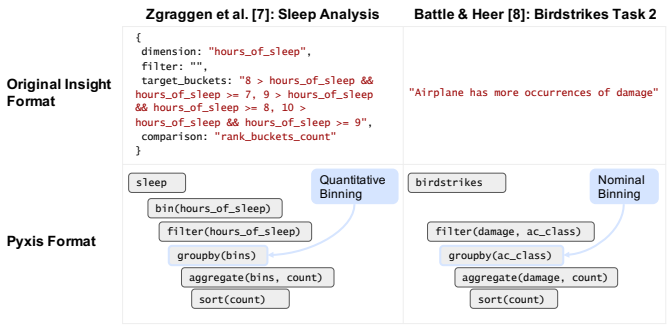


Fig. 7: Although the analytic knowledge observed by Zggraggen et al. [7] and Battle and Heer [8] have very different formats, they share striking structural similarities when implemented using our formalism.

5.4 Use Case 4: Comparing Insight Studies

In this use case, we seek to understand how existing insight-based studies define insight and application-driven definitions compare with existing theories. In other words: *how useful are existing theories for existing studies?*

5.4.1 A Tale of Two Studies

For the sake of space, we limited our analysis to two insight-based studies by Zggraggen et al. [7] and Battle and Heer [8]. We used Pyxis to implement all of the insights shared by Zggraggen et al. for their sleep study dataset exploration task, i.e., from Figure 3 of their paper [7]. We also implemented three insights observed by Battle and Heer as analysts explored wildlife strikes data [8]. Examples of insights from both studies are shown in Figure 7, using the original formats at the top (lightly edited to maximize readability) and the corresponding Pyxis data structures (in this case, data transformations) at the bottom.

5.4.2 These Studies Share Striking Similarities

Upon first impression, these insights may appear to be drastically different. However, we observed striking structural similarities. First, both studies *record insights as analytic knowledge only* and omit domain knowledge. Second, the vast *majority of analytic knowledge were data transformations only*, but some data relationships are also observed, e.g., linear correlations. Third, our implemented data structures were very similar between the two studies. For example, most data transformations involved filtering and/or grouping by an independent variable and aggregating a dependent variable to produce summary statistics. An example is shown in Figure 7, where Zggraggen et al. observe quantitative binning on the `hours_of_sleep` attribute and Battle and Heer observe nominal binning using the `ac_class` attribute. Quantitative binning requires an extra step to discretize `hours_of_sleep` into bins, but otherwise, *their data transformations have identical structures*.

5.4.3 Existing Theory Only Covers a Subset of "Insights"

In some cases, we see strong overlaps between observed analytic knowledge from these studies and existing theory papers. For example, the `rank_buckets_count` insight in Figure 7 bears a strong resemblance to North's rent distribution example [6] (see subsection 5.2). However, North's examples only provide a partial fit for the analytic knowledge observed by Zggraggen et al. [7] and Battle and Heer [8]. For example, North does not provide explicit examples of multivariate relationships. Definitions emphasizing domain knowledge would also be a poor fit (e.g., [19]) since domain knowledge is not a focus of these studies.

Instead, theories that emphasize analytic knowledge as a whole are more appropriate. For example, Yang et al. formalize data facts, which encompass both data transformations and data relationships [26]. Kandogan and Engelke provide an alternative definition of analytic knowledge based primarily on relational query patterns [31]. Similarly, Demiralp et al. define insights as "a strong manifestation of a distributional property of the data, such as strong correlation, tight clustering, low dispersion, and so on." [62]. However, neither paper cites these definitions. We find that many insight-based studies observe the discovery of both data transformations and data relationships during visual analysis sessions [15], [18], [31], [37]. We also observe a recurring theme: these studies tend to reference the more popular definitions of insight rather than the most relevant ones. This pattern reveals an important limitation to existing theory: it is difficult to identify and apply the most relevant theories to insight-based studies. Our formalism takes a step towards addressing these challenges by providing a framework for navigating existing theories of insight.

5.4.4 Benefits of the Formalism

Given the importance of data distributions as analytic knowledge, we could reuse the corresponding Pyxis objects as *templates* for extracting this knowledge from new datasets, e.g., for visualization recommendation [73]. Further, rather than limiting recommendations to quantitative binning (i.e., standard histograms), we can also apply qualitative binning as observed by Battle and Heer [8]. To use Pyxis objects as templates, we can replace the assigned attributes in the Pyxis objects (e.g., `hours_of_sleep`, `ac_class`) with equivalent attribute(s) from a target dataset. This approach also reveals the potential of Pyxis as a language for *taxonomizing* observed insights, thereby extending existing categorizations (subsection 2.2) with a corpus of exemplars amenable to quantitative meta-analysis.

6 Discussion and Future Work

Researchers largely agree on the *structure* of insights, i.e., their major building blocks, but not their *semantics*, i.e., how these building blocks are interpreted. We hone existing structural consistencies into a unified formalism to quantify the complexity and scope of observed knowledge. We highlight exciting future directions based on this work.

6.1 Declarative Specification of Insights

We observe a consistent progression from an unstructured recording of analytic knowledge towards *declarative specification* of analytic knowledge. This is exemplified in the progressions of building blocks proposed by North [13], Kandogan and Engelke [31], Zgraggen et al. [7], and Andrienko and Andrienko [49] as well as in related work by Suh et al. [74]. These works converge on expressing a user's interpretation of data in terms of *relational queries* rather than a series of logged system interactions or interface manipulations. An example from Zgraggen et al. is shown in Figure 7. This direction can also provide interesting opportunities to intersect with related areas, such as databases and programming languages research [75]. Still, there are clear limitations to this approach revealed by our formalism (see subsection 5.2). For example, relational query languages like SQL lack declarative representations for multivariate and univariate data relationships. We highlight this as a critical gap to be filled in future theories.

6.2 Supporting Domain Knowledge

We observed a recurring theme across insight-based theories and studies: they lack precise specifications for domain knowledge, hindering our community's ability to reason about user knowledge as a whole. For example, a correlation between two generic variables is meaningless without domain context, e.g., a correlation between certain drugs and patient health outcomes [37] or the incidence of crime and racial injustice in Baltimore [41]. This is not solely a visualization provenance issue. For example, recent research shows how visualization recommendation systems [60], [73] and visualization languages [76] can be hindered by their inability to incorporate domain and task context. Thus, incorporating user domain knowledge appears to be a consistent challenge within visualization research.

As a first step, **we observe a need for more precise methods for expressing user domain knowledge**, ideally at the theoretical level. In this way, we can capture user domain knowledge with equivalent precision as analytic knowledge, allowing our community to more accurately model user learning and knowledge building over time.

One possibility for refining representations of domain knowledge is to exploit formalisms from knowledge graphs, which could enable us to automate traditionally manual practices using graph-based algorithms, e.g., to quantify the depth and breadth of user knowledge (subsection 5.2), to detect patterns in knowledge acquisition [43], [31], or even to provide additional context to visualization tools such as for visualization recommendation tasks [77]. Further, knowledge graphs could support bias detection by detecting problematic links between domain and analytic knowledge, and analytic knowledge and conclusions that are drawn.

6.3 Sharing Insight Data

A critical second step is to **prioritize sharing of insight data.** For example, although Liu and Heer share their annotations for user insights, it is impossible to extract the specific data users were analyzing in imMens in tandem with these annotations [21]. Zgraggen et al. [7] and Kandogan and Engelke [31] derive insight specifications from their study data but only share a handful of examples as brief anecdotes in their papers. Similarly, Guo et al. only describe a few example insights from their study in their paper [29]. In contrast, best practices in visualization provenance encourage consistent tracking, sharing, and reuse of recorded system interaction logs (e.g., [4], [78], [8], [79]).

Insight corpora could open new avenues for visualization research. For example, recorded insights could be aggregated to form reusable insight *templates* to detect insights within new datasets. Insight datasets could also be used as unit tests for evaluating insight specifications, for example, by adapting the coverage and diversity measures proposed by Gathani et al. to apply to insight specifications [4]. Our formalism eases this burden somewhat by providing precise specifications for the core building blocks of insights. In this way, researchers can choose which building blocks to

focus on and use Pyxis to share observed instances of these building blocks from their user studies.

7 Conclusion

Reviewing the literature, we find that researchers seem to agree on the *structure* of visual analysis insights, i.e., their major building blocks, but not the *semantics* of insights, i.e., how these building blocks are interpreted. We propose a unified formalism that integrates multiple theoretical definitions of insight and contribute a toolkit called Pyxis for specifying insights using our formalism. We demonstrate how to use Pyxis to implement existing definitions of insight and compare the resulting structures. We find that current definitions fail to support rich specifications of domain knowledge and analytic knowledge, revealing exciting visualization research opportunities in domain knowledge representation, automated analytical reasoning, and collaborations with data management and data mining researchers.

Acknowledgements

This research was supported by grants #2141506, #2142977, and #2118201 from the National Science Foundation (NSF).

References

[1] J. Hullman, "The purpose of visualization is insight, not pictures: An interview with ben shneiderman," *ACM Interactions*, 2019.
[2] L. Battle, R. Chang, and M. Stonebraker, "Dynamic Prefetching of Data Tiles for Interactive Visualization," ser. SIGMOD '16. ACM, 2016, pp. 1363–1375.
[3] S. Monadjemi, R. Garnett, and A. Ottley, "Competing Models: Inferring Exploration Patterns and Information Relevance via Bayesian Model Selection," *IEEE TVCG*, pp. 1–1, 2020.
[4] S. Gathani, S. Monadjemi, A. Ottley, and L. Battle, "A grammar-based approach for applying visualization taxonomies to interaction logs," in *CGF*, vol. 41, no. 3. Wiley Online Library, 2022, pp. 489–500.
[5] M. Pohl, S. Wiltner, S. Miksch, W. Aigner, and A. Rind, "Analysing interactivity in information visualisation," *KI-Künstliche Intelligenz*, vol. 26, pp. 151–159, 2012.
[6] C. North, "Toward measuring visualization insight," *IEEE CG&A*, vol. 26, no. 3, pp. 6–9, May 2006.
[7] E. Zgraggen, Z. Zhao, R. Zeleznik, and T. Kraska, "Investigating the Effect of the Multiple Comparisons Problem in Visual Analysis," ser. CHI '18. ACM, Apr. 2018, pp. 1–12.
[8] L. Battle and J. Heer, "Characterizing Exploratory Visual Analysis: A Literature Review and Evaluation of Analytic Provenance in Tableau," *CGF*, vol. 38, no. 3, pp. 145–159, 2019.
[9] A. Rind, W. Aigner, M. Wagner, S. Miksch, and T. Lammarsch, "Task Cube: A three-dimensional conceptual space of user tasks in visualization design and evaluation," *Information Visualization*, vol. 15, no. 4, pp. 288–300, Oct. 2016, publisher: SAGE Publications.
[10] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer, "Vega-lite: A grammar of interactive graphics," *IEEE TVCG*, vol. 23, no. 1, pp. 341–350, 2017.
[11] A. M. McNutt and R. Chugh, *Integrated Visualization Editing via Parameterized Declarative Templates*, ser. CHI '21. ACM, 2021.
[12] L. Battle and A. Ottley, "What exactly is an insight? a literature review," *arXiv preprint arXiv:2307.06551 (to appear in IEEE VIS 2023 – Short Papers)*, 2023.
[13] C. North, P. Saraiya, and K. Duca, "A comparison of benchmark task and insight evaluation methods for information visualization," *Information Visualization*, vol. 10, no. 3, pp. 162–181, Jul. 2011.
[14] R. Chang, C. Ziemkiewicz, T. M. Green, and W. Ribarsky, "Defining Insight for Visual Analytics," *IEEE CG&A*, vol. 29, no. 2, pp. 14–17, Mar. 2009.
[15] P. Saraiya, C. North, and K. Duca, "An Insight-Based Methodology for Evaluating Bioinformatics Visualizations," *IEEE TVCG*, vol. 11, no. 4, pp. 443–456, Jul. 2005.
[16] ——, "An Evaluation of Microarray Visualization Tools for Biological Insight," in *IEEE Symposium on Information Visualization*, Oct. 2004, pp. 1–8, iSSN: 1522-404X.
[17] E. K. Choe, B. Lee, and m. c. schraefel, "Characterizing Visualization Insights from Quantified Selfers' Personal Data Presentations," *IEEE CG&A*, vol. 35, no. 4, pp. 28–37, Jul. 2015.
[18] M. Smuc, E. Mayr, T. Lammarsch, W. Aigner, S. Miksch, and J. Gärtner, "To Score or Not to Score? Tripling Insights for Participatory Design," *IEEE CG&A*, vol. 29, no. 3, pp. 29–38, May 2009.
[19] D. Gotz, M. X. Zhou, and V. Aggarwal, "Interactive Visual Synthesis of Analytic Knowledge," in *2006 IEEE Symposium On Visual Analytics Science And Technology*, Oct. 2006, pp. 51–58.
[20] Z. Pousman, J. Stasko, and M. Mateas, "Casual Information Visualization: Depictions of Data in Everyday Life," *IEEE TVCG*, vol. 13, no. 6, pp. 1145–1152, Nov. 2007.
[21] Z. Liu and J. Heer, "The Effects of Interactive Latency on Exploratory Visual Analysis," *IEEE TVCG*, vol. 20, no. 12, pp. 2122–2131, Dec. 2014.
[22] B. Karer, H. Hagen, and D. J. Lehmann, "Insight beyond numbers: The impact of qualitative factors on visual data analysis," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 1011–1021, 2021.
[23] E. Zgraggen, A. Galakatos, A. Crotty, J.-D. Fekete, and T. Kraska, "How progressive visualizations affect exploratory analysis," *IEEE TVCG*, vol. 23, no. 8, pp. 1977–1987, 2017.
[24] P. Saraiya, C. North, Vy Lam, and K. Duca, "An Insight-Based Longitudinal Study of Visual Analytics," *IEEE TVCG*, vol. 12, no. 6, pp. 1511–1522, Nov. 2006.
[25] C. Plaisant, J.-D. Fekete, and G. Grinstein, "Promoting Insight-Based Evaluation of Visualizations: From Contest to Benchmark Repository," *IEEE TVCG*, vol. 14, no. 1, pp. 120–134, Jan. 2008.
[26] Yang Chen, Jing Yang, and W. Ribarsky, "Toward effective insight management in visual analytics systems," in *2009 IEEE Pacific Visualization Symposium*, Apr. 2009, pp. 49–56, iSSN: 2165-8773.
[27] D. Sacha, A. Stoffel, F. Stoffel, B. C. Kwon, G. Ellis, and D. A. Keim, "Knowledge Generation Model for Visual Analytics," *IEEE TVCG*, vol. 20, no. 12, pp. 1604–1613, Dec. 2014.
[28] S. R. Gomez, H. Guo, C. Ziemkiewicz, and D. H. Laidlaw, "An insight- and task-based methodology for evaluating spatiotemporal visual analytics," in *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)*, Oct. 2014, pp. 63–72.
[29] H. Guo, S. R. Gomez, C. Ziemkiewicz, and D. H. Laidlaw, "A Case Study Using Visualization Interaction Logs and Insight Metrics to Understand How Analysts Arrive at Insights," *IEEE TVCG*, vol. 22, no. 1, pp. 51–60, Jan. 2016.
[30] A. Srinivasan, S. M. Drucker, A. Endert, and J. Stasko, "Augmenting Visualizations with Interactive Data Facts to Facilitate Interpretation and Communication," *IEEE TVCG*, vol. 25, no. 1, pp. 672–681, Jan. 2019.
[31] E. Kandogan and U. Engelke, "Towards a unified representation of insight in human-in-the-loop analytics: A user study," ser. HILDA'18. ACM, 2018.
[32] J. S. Yi, Y.-a. Kang, J. T. Stasko, and J. A. Jacko, "Understanding and characterizing insights: how do people gain insights using information visualization?" in *Proceedings of the 2008 conference on BEyond time and errors novel evaLuation methods for Information Visualization - BELIV '08*. Florence, Italy: ACM Press, 2008, p. 1.
[33] R. A. Amar and J. T. Stasko, "Knowledge precepts for design and evaluation of information visualizations," *IEEE TVCG*, vol. 11, no. 4, pp. 432–442, Jul. 2005.
[34] Y. B. Shrinivasan, D. Gotz, and J. Lu, "Connecting the dots in visual analysis," in *IEEE VAST*, Oct. 2009, pp. 123–130.
[35] Y. B. Shrinivasan and J. J. van Wijk, "Supporting the analytical reasoning process in information visualization," ser. CHI '08. ACM, Apr. 2008, pp. 1237–1246.
[36] D. Gotz and M. X. Zhou, "Characterizing Users' Visual Analytic Activity for Insight Provenance," *Information Visualization*, vol. 8, no. 1, pp. 42–55, Jan. 2009, publisher: SAGE Publications.
[37] C. He, L. Micallef, L. He, G. Peddinti, T. Aittokallio, and G. Jacucci, "Characterizing the Quality of Insight by Interactions: A Case Study," *IEEE TVCG*, pp. 1–1, 2020.
[38] T. M. Green, W. Ribarsky, and B. Fisher, "Visual analytics for complex concepts using a human cognition model," in *IEEE VAST*, Oct. 2008, pp. 91–98.
[39] W. Willett, J. Heer, J. Hellerstein, and M. Agrawala, "Commentspace: Structured support for collaborative visual analysis,"

in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '11. ACM, 2011, p. 3131–3140.
[40] W. Dou, D. H. Jeong, F. Stukes, W. Ribarsky, H. R. Lipford, and R. Chang, "Recovering Reasoning Processes from User Interactions," *IEEE CG&A*, vol. 29, no. 3, pp. 52–61, May 2009.
[41] A. Mathisen, T. Horak, C. N. Klokmose, K. Grønbæk, and N. Elmqvist, "InsideInsights: Integrating Data-Driven Reporting in Collaborative Visual Analytics," *CGF*, vol. 38, no. 3, pp. 649–661, 2019.
[42] W. A. Pike, J. Stasko, R. Chang, and T. A. O'Connell, "The Science of Interaction," *Information Visualization*, vol. 8, no. 4, pp. 263–274, Jan. 2009, publisher: SAGE Publications.
[43] A. Toniolo, F. Cerutti, T. J. Norman, N. Oren, J. A. Allen, M. Srivastava, and P. Sullivan, "Human-machine collaboration in intelligence analysis: An expert evaluation," *Intelligent Systems with Applications*, vol. 17, p. 200151, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2667305322000886
[44] M. Brehmer and T. Munzner, "A Multi-Level Typology of Abstract Visualization Tasks," *IEEE TVCG*, vol. 19, no. 12, pp. 2376–2385, Dec. 2013.
[45] H. Lam, M. Tory, and T. Munzner, "Bridging from Goals to Tasks with Design Study Analysis Reports," *IEEE TVCG*, vol. 24, no. 1, pp. 435–445, Jan. 2018.
[46] R. Amar, J. Eagan, and J. Stasko, "Low-level components of analytic activity in information visualization," in *INFOVIS 2005.*, Oct. 2005, pp. 111–117, iSSN: 1522-404X.
[47] Y. Kang and J. Stasko, "Examining the Use of a Visual Analytics System for Sensemaking Tasks: Case Studies with Domain Experts," *IEEE TVCG*, vol. 18, no. 12, pp. 2869–2878, Dec. 2012.
[48] J. Bertin, *Semiology of graphics*. University of Wisconsin press, 1983.
[49] N. Andrienko and G. Andrienko, *Exploratory analysis of spatial and temporal data: a systematic approach*. Springer Science & Business Media, 2006.
[50] E. F. Codd, "A relational model of data for large shared data banks," *Commun. ACM*, vol. 13, no. 6, p. 377–387, jun 1970.
[51] L. Wilkinson, "The grammar of graphics," in *Handbook of computational statistics*. Springer, 2012, pp. 375–414.
[52] W. Yun, X. Zhang, Z. Li, H. Liu, and M. Han, "Knowledge modeling: A survey of processes and techniques," *International Journal of Intelligent Systems*, vol. 36, no. 4, pp. 1686–1720, 2021.
[53] A. Kale, Y. Wu, and J. Hullman, "Causal support: Modeling causal inferences with visualizations," *IEEE TVCG*, vol. 28, no. 1, pp. 1150–1160, 2022.
[54] K. Xu, A. Ottley, C. Walchshofer, M. Streit, R. Chang, and J. Wenskovitch, "Survey on the Analysis of User Interactions and Visualization Provenance," *CGF*, vol. 39, no. 3, pp. 757–783, 2020.
[55] A. Hogan, E. Blomqvist, M. Cochez, C. D'amato, G. D. Melo, C. Gutierrez, S. Kirrane, J. E. L. Gayo, R. Navigli, S. Neumaier, A.-C. N. Ngomo, A. Polleres, S. M. Rashid, A. Rula, L. Schmelzeisen, J. Sequeda, S. Staab, and A. Zimmermann, "Knowledge graphs," *ACM Comput. Surv.*, vol. 54, no. 4, jul 2021.
[56] S. Ji, S. Pan, E. Cambria, P. Marttinen, and S. Y. Philip, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 2, pp. 494–514, 2021.
[57] C. Bizer, T. Heath, T. Berners-Lee *et al.*, "Linked data-the story so far," *IJSWIS*, vol. 5, no. 3, pp. 1–22, 2009.
[58] I. Nonaka and H. Takeuchi, "The knowledge-creating company," *Harvard business review*, vol. 85, no. 7/8, p. 162, 2007.
[59] N. Andrienko, T. Lammarsch, G. Andrienko, G. Fuchs, D. Keim, S. Miksch, and A. Rind, "Viewing Visual Analytics as Model Building," *CGF*, vol. 37, no. 6, pp. 275–299, 2018.
[60] R. Zehrung, A. Singhal, M. Correll, and L. Battle, "Vis ex machina: An analysis of trust in human versus algorithmically generated visualization recommendations," ser. CHI '21. ACM, 2021.
[61] L. Harrison, F. Yang, S. Franconeri, and R. Chang, "Ranking visualizations of correlation using weber's law," *IEEE TVCG*, vol. 20, no. 12, pp. 1943–1952, 2014.
[62] Demiralp, Çağatay and Haas, Peter J. and Parthasarathy, Srinivasan and Pedapati, Tejaswini, "Foresight: recommending visual insights," *Proc. VLDB Endow.*, vol. 10, no. 12, pp. 1937–1940, Aug. 2017.
[63] A. Ottley, R. Garnett, and R. Wan, "Follow The Clicks: Learning and Anticipating Mouse Interactions During Exploratory Data Analysis," *CGF*, vol. 38, no. 3, pp. 41–52, 2019.
[64] C. Stolte, D. Tang, and P. Hanrahan, "Polaris: a system for query, analysis, and visualization of multidimensional relational databases," *IEEE TVCG*, vol. 8, no. 1, pp. 52–65, 2002.
[65] B. Shneiderman, "The eyes have it: a task by data type taxonomy for information visualizations," in *Proceedings 1996 IEEE Symposium on Visual Languages*, Sep. 1996, pp. 336–343, iSSN: 1049-2615.
[66] E. Meijer, "The world according to linq," *CACM*, vol. 54, no. 10, pp. 45–51, 2011.
[67] B. Chandramouli, J. Goldstein, M. Barnett, R. DeLine, D. Fisher, J. C. Platt, J. F. Terwilliger, and J. Wernsing, "Trill: A high-performance incremental query processor for diverse analytics," *Proc. VLDB Endow.*, vol. 8, no. 4, p. 401–412, dec 2014.
[68] P. Pirolli and S. Card, "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, Jan. 2005.
[69] Vega Development Team, "Transforms — vega," https://vega.github.io/vega/docs/transforms/, 2023.
[70] J. Heer, "Arquero — arquero," https://uwdata.github.io/arquero/, 2021.
[71] S. Jain, D. Moritz, D. Halperin, B. Howe, and E. Lazowska, "Sqlshare: Results from a multi-year sql-as-a-service experiment," ser. SIGMOD '16. ACM, 2016, p. 281–293.
[72] T. McCabe, "A complexity measure," *IEEE Transactions on Software Engineering*, vol. SE-2, no. 4, pp. 308–320, 1976.
[73] Z. Zeng, P. Moh, F. Du, J. Hoffswell, T. Y. Lee, S. Malik, E. Koh, and L. Battle, "An evaluation-focused framework for visualization recommendation algorithms," *IEEE TVCG*, vol. 28, no. 1, pp. 346–356, 2022.
[74] A. Suh, Y. Jiang, A. Mosca, E. Wu, and R. Chang, "A grammar for hypothesis-driven visual analysis," *arXiv preprint arXiv:2204.14267*, 2022.
[75] L. Battle and C. Scheidegger, "A structured review of data management technology for interactive visualization and analysis," *IEEE TVCG*, vol. 27, no. 2, pp. 1128–1138, 2021.
[76] L. Battle, D. Feng, and K. Webber, "Exploring d3 implementation challenges on stack overflow," in *IEEE VIS*, 2022, pp. 1–5.
[77] H. Li, Y. Wang, S. Zhang, Y. Song, and H. Qu, "Kg4vis: A knowledge graph-based approach for visualization recommendation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 1, pp. 195–205, 2022.
[78] M. Feng, E. Peck, and L. Harrison, "Patterns and Pace: Quantifying Diverse Exploration Behavior with Visualizations on the Web," *IEEE TVCG*, vol. 25, no. 1, pp. 501–511, Jan. 2019.
[79] Z. Cutler, K. Gadhave, and A. Lex, "Trrack: A library for provenance-tracking in web-based visualizations," in *IEEE VIS*, 2020, pp. 116–120.

Leilani Battle is an Assistant Professor in the Paul G. Allen School for Computer Science & Engineering at the University of Washington. Her research focuses on developing interactive data-intensive systems that aid analysts in performing complex data exploration and analysis. She completed a postdoc in the UW IDL and earned a PhD degree from MIT in the MIT Database Group.

Alvitta Ottley is an Associate Professor in Computer Science & Engineering Department at Washington University in St. Louis, Missouri, USA. Her research uses interdisciplinary approaches to solve problems such as how best to display information for effective decision-making and how to design human-in-the-loop visual analytics interfaces that are more attuned to how people think. She earned a PhD degree from Tufts University in the VALT group.