Data analysis tools for statistical non-experts

Eunice Jun

A dissertation

submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington
2023

Reading Committee:
Jeffrey Heer, Co-Chair
René Just, Co-Chair
Tyler H. McCormick

Program Authorized to Offer Degree:
Paul G. Allen School of Computer Science & Engineering

© Copyright 2023

Eunice Jun

University of Washington

Abstract

Data analysis tools for statistical non-experts

Eunice Jun

Co-chairs of the Supervisory Committee:
Jerre D. Noe Endowed Professor Jeffrey Heer
Paul G. Allen School of Computer Science & Engineering
Associate Professor René Just
Paul G. Allen School of Computer Science & Engineering

Data analysis is critical to science, public policy, and business. Despite their importance, statistical analyses are difficult to author, especially for researchers with expertise outside of statistics. Existing statistical tools, prioritizing mathematical expressivity and computational control, are low-level while researchers' motivating questions and hypotheses are high-level. Researchers need to translate their questions and hypotheses into low-level statistical code in an error-prone process that involves grappling with their domain knowledge, statistics, and programming.

In this talk, I will introduce two tools that embody a new way of authoring analyses: Tea and Tisane. Researchers directly express their domain knowledge through higher level abstractions, and the tools will validate the data, select a statistical analysis, and implement it, all while educating analysts about why a statistical approach is valid. Tea helps analysts author statistical tests. Tea's key insight is that statistical test selection can be cast as a constraint satisfaction problem. Tisane enables analysts to author generalized linear models with or without mixed effects, which are difficult for even statistical experts to author. Using Tisane, analysts can express their conceptual models using a high-level domain specific language. Tisane translates these conceptual models into causal DAGs and engages analysts in a disambiguation process to arrive at an output statistical model. Real-world researchers have already used these tools to conduct analyses in published research that push their own disciplines forward. I will also introduce "hypothesis formalization," a series of cognitive and operational steps analysts take to translate their research questions into statistical implementations. Hypothesis formalization retrospectively explains why Tea improves statistical testing and directly inspired the design of Tisane.

Tea and Tisane serve as platforms for further research into computational support for statistical analysis. This talk also exemplifies how combining human-computer interaction with other areas in

and outside of computer science leads to software tools that impact real-world users.

Acknowledgements

Insert acknowledgments here.

DEDICATION

To all the wild women who have danced with me. I promise to never stop.

i stand
on the sacrifices
of a million women before me
thinking
what can i do
to make this mountain taller
so the women after me
can see farther
- legacy
Rupi Kaur

There is a special place in hell for women who don't help other women.
Madeleine Albright

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Statistical analysis plays a critical role in how people evaluate data and make decisions. Policy makers rely on models to track disease, inform health recommendations, and allocate resources. Scientists develop, evaluate, and compare theories based on statistical results. Journalists report on new findings in science, which individuals use to make decisions that impact their nutrition, finances, and other aspects of their lives. Faulty statistical models can lead to spurious estimations of disease spread, findings that do not generalize or reproduce, and a misinformed public.

Despite the prevalence of statistical analyses and their central importance to a number of disciplines, they remain challenging to author accurately. The key challenge in developing accurate statistical models lies not in a lack of access to mathematical tools, of which there are many (e.g., R Team et al. [2013], Python Sanner et al. [1999], SPSS SPSS [2021], and SAS Inc. [2021]), but in accurately applying them in conjunction with domain theory, data collection, statistical knowledge, and programming ability McElreath [2020]. Analysts must translate their implicit domain knowledge into statistical models that they can then implement and execute in code. However, this process—which requires disciplinary, statistical, and programming expertise—is out of reach for statistical non-experts who depend on accurate analyses, including many researchers.

Approach

This dissertation asks if separating the above concerns and incorporating automated reasoning in statistical software could benefit statistical non-experts. Towards this goal, I combine techniques from human-computer interaction, programming languages/software engineering, and statistics to (i) characterize the cognitive and operational steps to author statistical analyses and (ii) develop novel interactive systems that enable statistical non-experts to author valid analyses. As detailed below, I not only move between systems building and empirical studies but use each to deepen and enhance the other.

The work described in the dissertation demonstrates the following:

Thesis statement Domain-specific languages that provide abstractions for expressing conceptual knowledge, data collection procedures, and analysis intents instead of specific statistical modeling decisions coupled with automated reasoning to compile conceptual specifications into statistical analysis code help statistical non-experts more readily author valid analyses.

Three challenges fall out of this thesis statement:

Challenge 1: How to make implicit domain knowledge explicit.

Designing abstractions focused on conceptual knowledge requires identifying what domain knowledge analysts want and can express and balancing these constraints with what automated reasoning approaches may require. What is easy to express and what is easy to assume for the sake of automation may be at odds, especially when analysts provide ambiguous specifications that could be compiled into multiple statistical analyses. The challenge therefore, is to design language constructs that are usable for analysts and useful for automated reasoning and support interactive program specification as necessary.

Challenge 2: Represent and reason about key statistical analysis decisions

A central idea in this thesis is that software systems should take on the responsibility of translating conceptual knowledge into statistical analyses. This is akin to a compilation process that requires representing the conceptual knowledge analysts express and reasoning over it to derive statistical analyses that respect statistical best practices and rules. A major challenge is in picking representations so that the reasoning is straightforward.

Challenge 3: Increase analysts' statistical knowledge/understanding

While automating statistical analysis can be helpful, analysts relying on data to make high-impact decisions (e.g., policy, scientific discovery) often need to understand why an analysis approach is appropriate and what the implications of the results are to their domain. Furthermore, software can inform how users approach future analyses. Therefore, educating analysts about the applicability and impact of statistical decisions and guiding their interpretation of results are important.

Summary of Contributions

Add overview figure from reserach statement This dissertation makes systems and empirical contributions. Additionally, the process of designing and developing domain-specific languages for end-users shows the first steps towards developing methods for user-centric language design.

Specifically, I designed and implemented two systems, Tea Jun et al. [2019] and Tisane **?**, that leverage **domain-specific languages** (DSLs) to capture analysts' implicit assumptions and conceptual knowledge. Users **interactively compile** these high-level specifications into low-level code. To infer valid statistical analyses, the systems **programmatically represent and reason about core statistical authoring challenges** as constraints and graphs (**??**).

In summary, this dissertation's key contributions are

- a **formal constraint-based model** to specify and select among common Null Hypothesis Statistical Tests in Tea (see Chapter 3);
- empirical findings of how authoring analyses requires integrating conceptual, data, statistical, and programming expertise, which we summarize in our **theory of hypothesis formalization** (see Chapter 4);
- an analysis of how the current statistical software ecosystem does not explicitly support and may even hinder hypothesis formalization, suggesting new **design opportunities and implications** (see Chapter 4);
- a **mixed-initiative approach** for "interactively compiling" linear models from conceptual and data relationships in Tisane;
- empirical **findings on researchers' implicit semantics of conceptual models** (see Chapter 5);
- **new language constructs and interaction methods** for reflecting on and refining conceptual models in a second version of Tisane, which we call rTisane (see Chapter 5); and
- qualitative and quantitative **results showing the benefit of recording conceptual models and compiling them into statistical models** in rTisane over a scaffolded workflow (see Chapter 5).

Chapter 2

Related work

This dissertation builds on theories of sensemaking, empirical findings on current analytical praxis, and existing tools throughout the data lifecycle. Additionally, this dissertation uses Donald Campbell's theory of validity to motivate system designs and interpret evaluation results. Subsequent sections provide additional background as applicable.

2.1 Statistical data analysis as sensemaking

Human beings engage in *sensemaking* to acquire new knowledge. Several theories of sensemaking Pirolli and Card [2005]; Russell et al. [1993]; Klein et al. [2007] describe how and when human beings seek and integrate new data (e.g., observations, experiences, etc.) to develop their mental models about the world.

Russell et al. Russell et al. [1993] define sensemaking as "the process of searching for a representation and encoding data in that representation to answer task-specific questions." Russell et al. emphasize the importance of external representations. Sensemaking is the iterative process of searching for and refining external representations in a "learning loop complex" that involves transitioning back and forth between (i) searching for and (ii) instantiating representations. External representations are critical because they influence the quality of conclusions reached at the end of the sensemaking process and affect how much time and effort is required in the process. Some representations may lead to insights more quickly. Indeed, we posit and find that statistical analysis, specifically hypothesis formalization (Chapter 4), is a learning loop Russell et al. [1993] where the conceptual research question or hypothesis is an external representation of a set of assumptions analysts may have about the world (e.g., an implicit causal model), that ultimately affects which statistical models are implemented and which results are obtained. We also find that there are smaller learning loops—for revising explicit causal models, mathematical equations, and partially specified models—embedded in the larger loop of hypothesis formalization.

Grolemund and Wickham argued for statistical data analysis as a sensemaking activity Grolemund and Wickham [2014]. They emphasize the (1) bidirectional nature of updating mental models of the world and hypotheses based on data and collecting data based on hypotheses and (2) the process of identifying and reconciling discrepancies between hypotheses and data. Similar to Russell et al., Grolemund and Wickham's model demonstrates the importance of representing and re-representing conceptual knowledge. Grolemund and Wickham's theory of data analysis includes a back and forth between an analyst's "schema" of how a phenomenon occurs in the world, a statistical model, and data. Analysts' domain expertise influence their schemas, which represent conceptual knowledge about known and unknown causal mechanisms, for example. Analysts' conceptual schema directly inform their hypotheses, which are statistical predictions represented in statistical models. These statistical models are then compared to collected data, and any discrepancies between the data and hypothesis require analysts to re-examine and possibly update their statistical model, schema, or both. Extending Grolemund and Wickham's model, our work on hypothesis formalization differentiates between conceptual and statistical hypotheses and probes the phases an analyst must go through to encode a conceptual hypothesis into a statistical model.

Given the centrality of external representations of implicit conceptual knowledge to authoring statistical analyses that help analysts make sense of the world, we argue that our statistical software should focus on helping analysts to express their conceptual hypotheses and implicit domain knowledge. Through the development of two software systems, Tea (Chapter 3) and Tisane(Chapter 5), we explore *how* to design programming abstractions and *what* those abstractions should include in order for statistical non-experts to externalize their implicit conceptual knowledge about a domain.

2.2 Empirical accounts of data analysis practice

Data analysis involves a number of tasks that involve data discovery, wrangling, profiling, modeling, and reporting Kandel et al. [2012]. Extending the findings of Kandel et al. Kandel et al. [2012], both Alspaugh et al. Alspaugh et al. [2018] and Wongsuphasawat et al. Wongsuphasawat et al. [2019] propose exploration as a distinct task. Whereas Wongsuphasawat et al. argue that exploration should subsume discovery and profiling, Alspaugh et al. describe exploration as an alternative to modeling. The importance of exploration and its role in updating analysts' understanding of the data and their goals and hypotheses is of note, regardless of the precise order or set of tasks. Battle and Heer describe exploratory visual analysis (EVA), a subset of exploratory data analysis (EDA) where visualizations are the primary interfaces and outputs for exploring data, as encompassing both data-focused (bottom-up) and goal- or hypothesis-focused (top-down) investigations Battle and Heer [2019]. In Chapter 4, we found that (i) analysts explored their data before modeling and (ii) exploratory observations sometimes prompted conceptual shifts in hypotheses (bottom-up) but at other times were guided by hypotheses and only impacted statistical analyses (top-down). In this way, data exploration appears to be an important intermediate step in hypothesis formalization, blurring the lines between exploratory and confirmatory data analysis.

Decisions throughout analysis tasks can give rise to a "garden of forking paths" Gelman and Loken [2013], which compounds for meta-analyses synthesizing previous findings Kale et al. [2019]. Liu, Boukhelifa, and Eagan Liu et al. [2019b] proposed a broad framework that characterizes analysis alternatives using three different *levels of abstraction*: cognitive (e.g., shifts in conceptual hypotheses), artifact (e.g., choice in statistical tools), and execution (e.g., computational tuning). *Cognitive* alternatives involve more conceptual shifts and changes (e.g., mental models, hypotheses). *Artifact* alternatives pertain to tooling (e.g., which software is used for analysis?), model (e.g., what is the general mathematical approach?), and data choices (e.g., which dataset is used?). *Execution* alternatives are closely related to artifact alternatives but are more fine-grained programmatic decisions (e.g., hyperparameter tuning). We find that hypothesis formalization involves all three levels of abstraction and provide a more granular depiction of how these levels cooperate with one another (Chapter 4).

Moreover, Liu, Althoff, and Heer Liu et al. [2019a] identified numerous decision points throughout the data lifecycle, which they call *end-to-end analysis*. They found that analysts often revisit key decisions during data collection, wrangling, modeling, and evaluation. Liu, Althoff, and Heer also found that researchers executed and selectively reported analyses that were already found in prior work and familiar to the research community. The focus of this thesis is on how any single pass or iteration occurs. We approach this work from the perspective that by understanding a single iteration, we may be able to focus analysts on their iterations that are most substantial and impactful and eliminate a number of unnecessary iterations that arise due to mistakes in aligning conceptual and statistical concerns, which we found in our case studies (see Section 5.4).

Importantly, our work differs in (i) scope and (ii) method from prior work in HCI on data analysis practices. Whereas translating a research question or hypothesis into a statistical analysis has remained implicit in prior descriptions of data analysis, we explicate this specific process. Additionally, while previous researchers have relied primarily on post-analysis interviews with analysts, our lab study (Section 4.4) enables us to observe decision making during this process in-situ.

2.3 Tools for data analysis

The software ecosystem for data analysis is vibrant, with numerous programming languages, software packages, and graphical-first tools. A common limitation of existing software is its siloing of statistical specification from the conceptual and data collection details that inadvertently influence statistical analysis. In contrast, the systems in this dissertation explore ways to leverage implicit conceptual and data collection knowledge to derive statistical analyses. Below, we compare and contrast this dissertation with existing software for conceptual modeling, study design, and statistical specification.

2.3.1 Tools for conceptual modeling

For statistical experts, causal diagramming is a common approach to externalizing implicit concpetual models. For instance, Daggity Textor et al. [2011] supports authoring, editing, and formally analyzing causal graphs through code and a visual editor. The key limitation of Daggity is that it requires analysts to specify a formal causal graph, which statistical non-experts, including many domain experts, may not be able to do Suzuki et al. [2020]; Suzuki and VanderWeele [2018]; Velentgas et al. [2013]. In fact, an open challenge for causal reasoning and discovery is in getting domain experts to express their implicit knowledge in a way that can be formally represented and reasoned about. Our work on Tisane directly addresses this challenge. Moreover, even if analysts are able to express causal diagrams in Dagitty, Dagitty does not translate queries analysts may have about the causal diagram (i.e., research questions, hypotheses) into statistical models that could assess specific relationships of interest. Tisane also overcomes this limitation for a set of queries about average causal effect.

For experts comfortable working with causal diagrams, software packages such as marginaleffects

2.3.2 Tools for study design

Several domain-specific languages Sloane and Hardin [2017]; Bakshy et al. [2014], software packages Tanaka [2021]; Blair et al. [2019], and standalone applications Mackay et al. [2007]; Eiselmayer et al. [2019] specialize in experiment design. A primary focus is to provide researchers low-level control over trial-level and randomization details. For example, JsPsych De Leeuw [2015] gives researchers fine-grained control over the design and presentation of stimuli for online experiments. At a mid-level of abstraction, Touchstone Mackay et al. [2007] is a tool for designing and launching online experiments. It also refers users to R and JMP for data analysis but does not help users author an appropriate statistical model. Touchstone2 Eiselmayer et al. [2019] helps researchers design experiments based on statistical power. At a high-level of abstraction, `edibble` Tanaka [2021] helps researchers plan their data collection schema. `edibble` aims to provide a “grammar of study design” that focuses users on their experimental manipulations in relation to specific units (e.g., participants, students, schools), the frequency and distribution of conditions (e.g., within-subjects vs. between-subjects), and measures to collect (e.g., age, grade, location) in order to output a table to fill in during data collection. While Tisane’s study design specification language uses an abstraction level comparable to `edibble`, Tisane is focused on using the expressed data measurement relationships to infer a statistical model. Additionally, Tisane’s SDSL provides conceptual relationships that are out of the scope of `edibble` but important for specifying conceptually valid statistical models.

2.3.3 Tools for statistical specification

A contribution of this thesis is a closer examination of how existing statistical analysis tools fail to support authoring (Section 4.5). Here, we contrast the systems developed in this thesis to discipline-specific software tools for research and more general automated statistics approaches.

Research has introduced tools to support statistical analysis in diverse domains. ExperiScope Guimbretière et al. [2007] supports users in analyzing complex data logs for interaction techniques. ExperiScope surfaces patterns in the data that would be difficult to detect manually and enables researchers to collect noisier data in the wild that have greater external validity. Statsplorer Wacharamanotham et al. [2015] is an educational web application for novices learning about statistics. While more focused on visualizing various alternatives for statistical tests, Statsplorer automates test selection (for a limited number of statistical tests and by executing simple switch statements) and the checking of assumptions (though it is currently limited to tests of normality and equal variance). Wacharamanotham et al. [2015] found that Statsplorer helps HCI students perform better in a subsequent statistics lecture. Similar in scope to Statsplorer, Tea is designed to help statistical non-experts author Null-Hypothesis Significance Tests. Tea supports twice as many statistical tests as Statsplorer, suggesting that Tea’s constraint-based approach is more expressive than Statsplorer’s decision-tree implementation for statistical test selection. In contrast to the above systems, a key design consideration for Tea and Tisane has been their ability to apply widely across disciplines and integrate into many existing workflows. Therefore, the systems are implemented as embedded DSLs in Python and R, two widely used programming languages for data science.

The Automatic Statistician Lloyd et al. [2014] generates a report listing all “interesting” relationships (e.g., correlations, statistical models, etc.). Although apparently complete, the Automatic Statistician may overlook analyses that are conceptually interesting and difficult, if not impossible, to deduce from data alone. Furthermore, AutoML tools such as Auto-WEKA Thornton et al. [2013], auto-sklearn Feurer et al. [2015], and H2O AutoML LeDell and Poirier [2020] also prioritize finding patterns in data and aim to make statistical methods more widely usable. However, Tea and Tisane differ from AutoML efforts in their researchers developing scientific theories. As a result, Tisane provides focus on analysts who prioritize explanation, not just prediction, such as support for specifying GLMMs, which some prominent AutoML tools, such as auto-sklearn Feurer et al. [2015], omit.

2.4 Validity in statistical data analysis

Finally, a aspect of this thesis is that software with conceptually grounded programming abstractions and automated reasoning can improve the validity of analyses. There are many working definitions of "validity," from predictive accuracy to a quality of how well experiments are designed to a trade-off between model simplicity and fit (e.g., R-squared). Donald Campbell's theory of validity Shadish [2010], widely adopted across disciplines, provides a framework for reasoning about and unifying many intuitive definitions of validity. Campbell defines four dimensions of validity: internal validity, external validity, statistical conclusion validity, and construct validity. This thesis focuses on enhancing statistical conclusion, external, and internal validity through the correct application and specification of statistical analyses that match analysts' intentions (i.e., their research questions and hypotheses) and data collection procedures. We do not address construct validity because construct validity is specific to a discipline's theories and is often debated over a relatively long period of time. In the conclusion (Chapter 6), we pick back up on opportunities to address construct validity through the applica

Chapter 3

Tea: A Domain-Specific Language and Runtime System for Hypothesis Testing

Remove "this paper"; Rephrase the language describing code and data releases.

> *The enormous variety of modern quantitative methods leaves researchers with the non-trivial task of matching analysis and design to the research question.*
>
> *- Ronald Fisher Fisher [1937]*

Since the development of modern statistical methods (e.g., Student's t-test, ANOVA, etc.), statisticians have acknowledged the difficulty of identifying which statistical tests people should use to answer their specific research questions. Almost a century later, choosing appropriate statistical tests for evaluating a hypothesis remains a challenge. As a consequence, errors in statistical analyses are common Kaptein and Robertson [2012], especially given that data analysis has become a common task for people with little to no statistical expertise.

A wide variety of tools (such as SPSS Wikipedia contributors [2019d], SAS Wikipedia contributors [2019c], and JMP Wikipedia contributors [2019a]), programming languages (e.g., R Wikipedia contributors [2019b]), and libraries (including numpy Oliphant [2006], scipy Jones et al. [2021a], and statsmodels **?**), enable people to perform specific statistical tests, but they do not address the fundamental problem that users may not know which statistical test to perform and how to verify that specific assumptions about their data hold. In fact, all of these tools place the burden of valid, replicable statistical analyses on the user and demand deep knowledge of statistics.

Users not only have to identify their research questions, hypotheses, and domain assumptions, but also must select statistical tests for their hypotheses (e.g., Student's t-test or one-way ANOVA). For each statistical test, users must be aware of the statistical assumptions each test makes about the data (e.g., normality or equal variance between groups) and how to check for them, which requires additional statistical tests (e.g., Levene's test for equal variance), which themselves may demand further assumptions about the data. This cognitively demanding process requires significant knowledge about statistical tests and their preconditions as well as the ability to perform the tests and verify their preconditions. This process can easily lead to mistakes.

In response, we design and developed Tea[1], a high-level declarative language for automating statistical test selection and execution that abstracts the details of statistical analysis from the users. Tea captures users' hypotheses and domain knowledge, translates this information into a constraint satisfaction problem, identifies all valid statistical tests to evaluate a hypothesis, and executes the tests. Tea's higher-level, declarative nature aims to lower the barrier to valid, replicable analyses.

Tea is easy to integrate directly into common data analysis workflows for users who have minimal programming experience. Tea is implemented as an open-source Python library, so programmers can use Tea wherever they use Python, including within Python notebooks.

In addition, Tea is flexible. Its abstraction of the analysis process and use of a constraint solver to select tests is designed to support its extension to emerging statistical methods, such as Bayesian analysis. Currently, Tea supports frequentist Null Hypothesis Significance Testing (NHST).

This work makes the following contributions:

- a novel DSL for automatically selecting and executing statistical analyses based on users' hypotheses and domain knowledge (subsection 3.3.2),
- a runtime system that formulates statistical test selection as a maximum constraint satisfaction problem (subsection 3.3.3), and
- an initial evaluation showing that Tea can express and execute common NHST statistical tests (Section 3.3.3).

After describing related work, the chapter describes a usage scenario, providing an overview of Tea (Section 3.3). Then, we discuss the concerns about statistics in the HCI community that shaped Tea's design (**??**), the implementation of Tea's programming language (subsection 3.3.2), the implementation of Tea's runtime system (subsection 3.3.3), and the evaluation of Tea as a whole (Section 3.3.3). The chapter concludes with the key limitations of this work and a discussion of how Tea demonstrates the feasibility of my approach central to the thesis statement(Section 1)

3.1 Background and Related work

Tea extends prior work on domain-specific languages for the data life cycle, tools for statistical analysis, and constraint-based approaches in HCI.

[1]named after Fisher's "Lady Tasting Tea" experiment Fisher [1937]

3.1.1 Domain-specific Languages for the Data Life Cycle

Prior domain-specific languages (DSLs) have focused on several different stages of data exploration, experiment design, and data cleaning to shift the burden of accurate processing from users to systems. To support data exploration, Vega-lite Satyanarayan et al. [2017] is a high-level declarative language that supports users in developing interactive data visualizations without writing functional reactive components. PlanOut Bakshy et al. [2014] is a DSL for expressing and coordinating online field experiments. More niche than PlanOut, Touchstone2 provides the Touchstone Language for specifying condition randomization in experiments (e.g., Latin Squares) Eiselmayer et al. [2019].essential aspect of the domain knowledge users encode in Tea programs. To support rapid data cleaning, Wrangler Kandel et al. [2011] combines a mixed-initiative interface with a declarative transformation language. Tea can be integrated with tools such as Wrangler that produce cleaned CSV files ready for analysis.

In comparison to these previous DSLs, Tea provides a language to support another crucial step in the data life cycle: statistical analysis.

3.1.2 Constraint-based Systems in HCI

Languages provide semantic structure and meaning that can be reasoned about automatically. For domains with well defined goals, constraint solvers can be a promising technique. Some of the previous constraint-based systems in HCI have been Draco Moritz et al. [2019] and SetCoLa Hoffswell et al. [2018], which formalize visualization constraints for graphs. Whereas SetCoLa is specifically focused on graph layout, Draco formalizes visualization best practices as logical constraints to synthesize new visualizations. The knowledge base can grow and support new design recommendations with additional constraints.

Another constraint-based system is Scout Swearngin et al. [2018], a mixed-initiative system that supports interface designers in rapid prototyping. Designers specify high-level constraints based on design concepts (e.g., a profile picture should be more emphasized than the name), and Scout synthesizes novel interfaces. Scout also uses Z3's theories of booleans and integer linear arithmetic.

We extend this prior work by providing the first constraint-based system for statistical analysis.

3.2 Statistical scope

Tea is designed for statistical tests common to Null Hypothesis Significance Testing (NHST). While there are calls to incorporate other methods of statistical analysis Kay et al. [2016]; Kaptein and Robertson [2012], Null Hypothesis Significance Testing (NHST) remains the norm in HCI and other disciplines. Therefore, Tea currently implements a module for NHST with the tests found to be most common by Wacharamanotham et al. [2015]. In particular, Tea supports four classes of tests: correlation (parametric: Pearson's r, Pointbiserial; non-parametric: Kendall's τ, Spearman's ρ), bivariate mean comparison (parametric: Student's t-test, Paired t-test; non-parametric: Mann-Whitney U, Wilcoxon signed rank, Welch's), multivariate mean comparison (parametric: F-test, Repeated measures one way ANOVA, Factorial ANOVA, Two-way ANOVA; non-parametric: Kruskal Wallis, Friedman), and comparison of proportions (Chi Square, Fisher's Exact). Tea also supports an implementation of bootstrapping Efron [1992].

3.3 Usage Scenario

This section describes how an analyst who has no statistical background can use Tea to answer their research questions. We use as an example analyst a historical criminologist who wants to determine how imprisonment differed across regions of the US in 1960[2]. Figure 3.1 shows the Tea code for this example.

The analyst specifies the data file's path in Tea. Tea handles loading and storing the data set for the duration of the analysis session. The analyst does not have to worry about reformatting the data during the analysis process in any way.

The analyst asks if the probability of imprisonment was higher in southern states than in non-southern states. The analyst identifies two variables that could help them answer this question: the probability of imprisonment ('Prob') and geographic location ('So'). Using Tea, the analyst defines the geographic location as a dichotomous nominal variable where '1' indicates a southern state and '0' indicates a non-southern state, and indicates that the probability of imprisonment is a numeric data type (ratio) with a range between 0 and 1.

The analyst then specifies their study design, defining the study type to be "observational study" (rather than "experimental study") and defining the contributor (independent) variable to be the geographic location and the outcome (dependent) variable to be the probability of imprisonment.

Based on their prior research, the analyst knows that the probability of imprisonment in southern and non-southern states is normally distributed. The analyst provides an assumptions clause to Tea in which they specify this domain knowledge. They also specify an acceptable Type I error rate (probability of finding a false positive result), more colloquially known as the 'significance threshold' ($\alpha = .05$) that is acceptable in criminology. If the analyst does not have assumptions or forgets to provide assumptions, Tea will use the default of $\alpha = .05$.

The analyst hypothesizes that southern states will have a higher probability of imprisonment than non-southern states. The analyst directly expresses this hypothesis in Tea. *Note that at no*

[2]The example is taken from Ehrlich Ehrlich [1973] and Vandaele Vandaele [1987]. The data set comes as part of the MASS package in R.

```
import tea
tea.data('UScrime.csv')                                          1
variables = [
    {
        'name' : 'So',
        'data type' : 'nominal',
        'categories' : ['0', '1']
    },
    {
        'name' : 'Prob',
        'data type' : 'ratio',
        'range' : [0,1]
    }
]
tea.define_variables(variables)                                  2
study_design = {
                  'study type': 'observational study',
                  'contributor variables': 'So',
                  'outcome variables': 'Prob',
               }
tea.define_study_design(study_design)
                                                                 3
assumptions = {
    'groups normally distributed': [['So', 'Prob']],
    'Type I (False Positive) Error Rate': 0.05
}
tea.assume(assumptions)                                          4
hypothesis = 'So:1 > 0'
tea.hypothesize(['So', 'Prob'], hypothesis)                      5
```

Figure 3.1: Sample Tea program. The specification outlines an experiment to analyze the relationship between geographic location ('So') and probability of imprisonment ('Prob') in a common USCrime data set Venables and Ripley [2013]; Kabacoff [2011]. See Section 3.3 for an explanation of the code. Tea programs specify 1) data, 2) variables, 3) study design, 4) assumptions, and 5) hypotheses.

point does the analyst indicate which statistical tests should be performed.

From this point on, Tea operates entirely automatically. When the analyst runs their Tea program, Tea checks properties of the data and finds that the Student's t-test is appropriate. Tea executes the Student's t-test and non-parametric alternatives, such as the Mann-Whitney U test, which provide alternative, consistent results.

Tea generates a table of results from executing the tests, ordered by their power (i.e., results from the parametric t-test will be listed first given that it has higher power than the non-parametric equivalent). Based on this output, the analyst concludes that their hypothesis—that the probability of imprisonment was higher in southern states than in non-southern states in 1960—is supported. The results from alternative statistical tests support this conclusion, so the analyst can be confident in their assessment.

The analyst can now share their Tea program with colleagues. Other researchers can easily see what assumptions the analyst made and what the intended hypothesis was (since these are explicitly stated in the Tea program), and reproduce the exact results using Tea.

3.3.1 System overview

Tea consists of a high-level programming language and a runtime system. There are three key steps to compiling a Tea program from user specifications to executing statistical analyses:

1. **Check for completeness and syntax.** Tea first checks that a user's program specifies a data set, variable declarations, study design description, a set of assumptions, and hypotheses using the correct syntax. For pre-registration, the data set can be empty (with only column names). If there are any syntax errors or missing parts, Tea will issue an error and stop execution.

2. **Check for consistent, well-formed hypotheses.** Using the variable declarations, Tea then checks that the hypotheses the user states are consistent with variable data types. For instance, Tea would issue an error and halt execution if a nominal variable was hypothesized to have a positive relationship with another nominal variable. If the nominal variables have categories given by numbers (e.g., a variable for education where '1' stands for 'High School', '2' for 'College', etc.), a linear relationship would be possible to compute by treating the categories as raw continuous values. However, treating the numbers as values is incorrect and the results misleading because the numbers represent discrete categories, not continuous values. Tea avoids such mistakes.

3. **Inspect data properties and infer valid statistical tests.** Once Tea's compiler verifies that a Tea program is complete, syntactically correct, and consistent, Tea's runtime sys-

tem inspects the data to verify properties about it and find a set of valid statistical tests. The higher-level Tea program is then compiled to logical constraints, which is further discussed in subsection 3.3.3.

3.3.2 Tea's Domain-Specific Programming Language

Tea is a domain-specific language embedded in Python. It takes advantage of existing Python data structures (e.g., classes, dictionaries, and enums). We chose Python because of its widespread adoption in data science. Tea is itself implemented as a Python library[3].

A key challenge in describing studies is determining the level of granularity necessary to produce an accurate analysis. In Tea programs, users describe their studies in five ways: (1) providing a data set, (2) describing the variables of interest in that data set, (3) specifying their study design, (4) stating their assumptions about the variables, and (5) formulating hypotheses about the relationships between variables. Figure 3.2 shows an an exaample Tea program and its output.

Data

Data is required for executing statistical analyses. One challenge in managing data for analysis is minimizing both duplicated data and user intervention.

To reduce the need for user intervention for data manipulation, Tea requires the data to be a CSV in long format. CSVs are a common output format for data storage and cleaning tools. Long format (sometimes called "tidy data" Wickham et al. [2014]) is a denormalized format that is widely used for collecting and storing data, especially for within-subjects studies.

Unlike R and Python libraries such as numpy Oliphant [2006], Tea only requires one instance of the data. Users do not have to duplicate the data or subsets of it for analyses that require the data to be in slightly different forms. Minimizing data duplication or segmentation is also important to avoid user confusion about where some data exist or which subsets of data pertain to specific statistical tests.

Optionally, users can also indicate a column in the data set that acts as a relational (or primary) key, or an attribute that uniquely identifies rows of data. For example, this key could be a participant identification number in a behavioral experiment. A key is useful for verifying a study design, described below. Without a key, Tea's default is that all rows in the data set comprise independent observations (that is, all variables are between subjects).

For pre-registration where there is no data, a CSV with only column names is necessary.

[3]Tea is open-source and available for download on `pip`, a common Python package manager.

Variables

Variables represent columns of interest in the data set. Variables have a name, a data type (*nominal*, *ordinal*, *interval*, or *ratio*), and, when appropriate, valid categories. Users (naturally) refer to variables through a Tea program using their names. Only nominal and ordinal variables have a list of possible categories. For ordinal variables, the categories are also ordered from left to right.

Variables encapsulate queries. The queries represent the index of the variable's column in the original data set and any filtering operations applied to the variable. For instance, it is common to filter by category for nominal variables.

Study Design

Three aspects of study design are important for conducting statistical analyses: (1) the type of study (observational study vs. randomized experiment), (2) the independent and dependent variables, and (3) the number of observations per participant (e.g., between-subjects variables vs. within-subjects variables).

For semantic precision, Tea uses different terms for independent and dependent variables for observational studies and experiments. In experiments, variables are described as either "independent" or "dependent" variables. In observational studies, variables are either "contributor" (independent) or "outcome" (dependent) variables.

Assumptions

Users' assumptions based on domain knowledge are critical for conducting and contextualizing studies and analyses. Often, users' assumptions are particular to variables and specific properties (e.g., equal variances across different groups). Current tools generally do not require that users encode these assumptions, leaving them implicit.

Tea takes the opposite approach to contextualize and increase the transparency of analyses. It requires that users be explicit about assumptions and statistical properties pertaining to the analysis as a whole (e.g., acceptable Type I error rate/significance threshold) and the data.

Tea supports two modes for treating user assumptions: *strict* and *relaxed*. In both modes, Tea verifies all user assumptions and issues warnings for assumptions that statistical testing does not verify. In the *strict* mode, Tea overrides user assumptions when selecting valid statistical tests. In the *relaxed* mode, Tea defers to user assumptions and proceeds as if the assumptions verified even if they did not. The *strict* mode is the default, but users can specify the *relaxed* mode. Figure 3.2 shows the two modes and the different warnings and output they generate.

If users also know that a data transformation (i.e., log transformation) applies to a variable, they can express this as an assumption. Data transformations are not properties to be verified but rather treatments of data that are applied during assumption verification, statistical test selection, and test execution, which is why they are included in the assumptions clause. The next section discusses the verification process for assumptions in greater detail.

Hypotheses

Hypotheses drive the statistical analysis process. Users often have hypotheses that are technically alternative hypotheses.

Tea focuses on capturing users' alternative hypotheses about the relationship between two or more variables. Tea uses the alternate hypothesis to conduct either a two-sided or one-sided statistical test. By default, Tea uses the null hypothesis that there is no relationship between variables.

3.3.3 Tea's Runtime System

Tea compiles programs into logical constraints about the data and variables, which it resolves using a constraint solver. A significant benefit of using a constraint solver is extensibility. Adding new statistical tests does not require modifying the core of Tea's runtime system. Instead, defining a new test requires expressing a single new logical relationship between a test and its preconditions.

At runtime, Tea invokes a solver that operates on the logical constraints it computes to produce a list of valid statistical tests to conduct. This process presents three key technical challenges: (1) incorporating statistical knowledge as constraints, (2) expressing user assumptions as constraints, and (3) recursively selecting statistical tests to verify preconditions of other statistical tests.

SMT Solver

As its constraint solver, Tea uses Z3 De Moura and Bjørner [2008], a Satisfiability Modulo Theory (SMT) solver.

Satisfiability is the process of finding an assignment to variables that makes a logical formula true. For example, given the logical rules $0 < x < 100$ and $y < x$, $\{x = 1, y = 0\}$, $\{x = 10, y = 5\}$, and $\{x = 99, y = -100\}$ would all be valid assignments that satisfy the rules. SMT solvers determine the satisfiability of logical formulas, which can encode boolean, integer, real number, and uninterpreted function constraints over variables. SMT solvers can also be used to encode constraint systems, as we use them here. A wide variety of applications ranging from the synthesis of novel interface designs Swearngin et al. [2018], the verification of website accessibility Panchekha et al. [2018], and the synthesis of data structures Loncaric et al. [2016] employ SMT solvers.

Logical Encodings

The first challenge of framing statistical test selection as a constraint satisfaction problem is defining a logical formulation of statistical knowledge.

Tea encodes the applicability of a statistical test based on its preconditions. A statistical test is applicable if and only if all of its preconditions (which are properties about variables) hold. We derived preconditions for tests from an online HCI and statistics course Klemmer and Wobbrock [2019], a statistics textbook Field et al. [2012], and publicly available data science resources from universities Bruin [2019]; Libraries [2019].

Tea represents each precondition for a statistical test as an uninterpreted function representing a property over one or more variables. Each property is assigned `true` if the property holds for the variable/s; similarly, if the property does not hold, the property function is assigned `false`.

Tea also encodes statistical knowledge about variable types and properties that are essential to statistical analysis as axioms, such as the constraint that only a continuous variable can be normally distributed.

Algorithm

Tea frames the problem of finding a set of valid statistical tests as a maximum satisfiability (MaxSAT) problem that is seeded with user assumptions.

First, Tea translates each user assumption about a data property into an axiom about a property and variable. As described in Section 3.3.2, user assumptions about properties but not data transformations are always checked. In the *strict* mode, Tea overrides any user assumptions it does not find to hold, creating an axiom that a property is `false`. In the *relaxed* mode, Tea defers to user assumptions, creating axioms that a property is `true`. For any user assumptions that do not pass statistical testing, Tea warns the user and explains how it will proceed depending on the mode.

Then, for each new statistical test Tea tries to satisfy, Tea checks to see if each precondition holds. For each precondition checked, Tea adds the property and variable checked as an axiom to observe as future tests are checked. If any property violates the axioms derived from users' assumptions, the property is removed and the test is invalidated. Users' assumptions always take precedence.

The constraint solver then prunes the search space. Tea does not compute all properties for all variables, a significant optimization when analyzing very large data sets.

At the end of this process, Tea finds a set of valid statistical tests to execute. If this set is empty, Tea defaults to its implementation of bootstrapping Efron [1992]. Otherwise, Tea proceeds and executes all valid statistical tests. Tea returns a table of results to users, applying multiple comparison corrections Holm [1979] and calculating effect sizes when appropriate.

Optimization: Recursive Queries

When Tea verifies a property holds for a variable, it often must invoke another statistical test. For example, to check that two groups have equal variance, Tea must execute Levene's test. The statistical test used for verification may then itself have a precondition, such as a minimum sample size.

Such recursive queries are inefficient for SMT solvers like Z3 to reason about. To eliminate recursion, Tea lifts some statistical tests to properties. For instance, Tea does not encode the Levene's test as a statistical test. Instead, Tea encodes the property of having equal variance between groups and executes the Levene's test for two groups when verifying that property for particular variables.

User Output

The result of running a Tea program with data is a listing of the results of executing valid statistical tests, as shown in Figure 3.2. For each valid statistical test executed, the output contains the properties of data that Tea checked and used to determine that a statistical test applied, the test statistic value, p-value (and an adjusted p-value, if applicable), effect sizes (Cohen's d Cohen [1988] and Vargha Delaney A12 Vargha and Delaney [2000]), the alpha level the user specified in their program, the precise null hypothesis the statistical test examined, an interpretation of the results in APA format Association et al. [1983], and text recommending users to focus on effect size rather than the p-value for a holistic view of their data. This output is intended to inform users of why Tea selected specific statistical tests and how to interpret their results.

Why constraints? are they really necessary?

Initial Evaluation

We assessed the validity of Tea in two ways. First, we compared Tea's suggestions of statistical tests to suggestions in textbook tutorials. We use these tutorials as a proxy for expert test selection. Second, for each tutorial, we compared the analysis results of the test(s) suggested by Tea to those of the test suggested in the textbook as well as all other candidate tests. We use the set of all candidate tests as as a proxy for non-expert test selection.

We differentiate between *candidate* and *valid* tests. A candidate test can be computed on the data, when ignoring any preconditions regarding the data types or distributions. A valid test is a candidate test for which all preconditions are satisfied.

3.3.4 How does Tea compare to textbook tutorials?

Our goal was to compare Tea to expert recommendations.

Table 3.1: Results of applying Tea to 12 textbook tutorials.

Tea is comparable to an expert selecting statistical tests. Tea can prevent false positive and false negative results by suggesting only tests that satisfy all assumptions. *Tutorial* gives the test described in the textbook; *Candidate tests (p-value)* gives all tests a user could run on the provided data with corresponding p-values; *Assumptions* gives all satisfied (lightly shaded) and violated (white) assumptions; *Tea suggests* indicates which tests Tea suggests based on their preconditions (assumptions about the data). **Emphasized** p-values indicate instances where a candidate test leads to a wrong conclusion about statistical significance. Although this table focuses on p-values, Tea produces richer output that provides a more holistic view of the statistical analysis results by including effect sizes, for instance. Refer to Figure 3.2 for an example of output from a Tea program.

Tutorial	Candidate tests (p-value)		Assumptions*	Tea suggests
Pearson Kabacoff [2011]	Pearson's r	(6.96925e-06)	② ④ ⑤	—
	Kendall's τ	(2.04198e-05)	② ④	✓
	Spearman's ρ	(2.83575e-05)	② ④	✓
Spearman's ρ Field et al. [2012]	Spearman's ρ	(.00172)	② ④	✓
	Pearson's r	(.01115)	② ④	—
	Kendall's τ	(.00126)	② ④	✓
Kendall's τ Field et al. [2012]	Kendall's τ	(.00126)	② ④	✓
	Pearson's r	(.01115)	② ④	—
	Spearman's ρ	(.00172)	② ④	✓
Pointbiserial Field et al. [2012]	Pointbiserial (Pearson's r)	(.00287)	② ④ ⑤	—
	Spearman's ρ	(.00477)	② ④	—
	Kendall's τ	(.00574)	② ④	—
	Bootstrap	(<0.05)		✓
Student's t-test Kabacoff [2011]	Student's t-test	(.00012)	② ④ ⑤ ⑥ ⑦ ⑧	✓
	Mann-Whitney U	(9.27319e-05)	② ④ ⑦ ⑧	✓
	Welch's t-test	(.00065)	② ④ ⑤ ⑦ ⑧	✓
Paired t-test Field et al. [2012]	Paired t-test	(.03098)	② ④ ⑤ ⑦ ⑧	✓
	Student's t-test	(**.10684**)	② ④ ⑤ ⑦	—
	Mann-Whitney U	(**.06861**)	② ④ ⑦	—
	Wilcoxon signed rank	(.04586)	② ④ ⑦ ⑧	✓
	Welch's t-test	(**.10724**)	② ⑦	—
Wilcoxon signed rank Field et al. [2012]	Wilcoxon signed rank	(.04657)	② ④ ⑦ ⑧	✓
	Student's t-test	(.02690)	② ④ ⑦	—
	Paired t-test	(.01488)	② ④ ⑤ ⑦ ⑧	—
	Mann-Whitney U	(.00560)	② ④ ⑦	—
	Welch's t-test	(.03572)	② ④ ⑦	—
F-test Field et al. [2012]	F-test	(9.81852e-13)	② ④ ⑤ ⑥ ⑨	✓
	Kruskal Wallis	(2.23813e-07)	② ④ ⑨	✓
	Friedman	(8.66714e-07)	② ⑦	—
	Factorial ANOVA	(9.81852e-13)	② ④ ⑤ ⑥ ⑨	✓
Kruskal Wallis Field et al. [2012]	Kruskal Wallis	(.03419)	② ④ ⑨	✓
	F-test	(**.05578**)	② ④ ⑤ ⑨	—
	Friedman	(3.02610e-08)	② ⑦	—
	Factorial ANOVA	(**.05578**)	② ④ ⑤ ⑨	—
Repeated measures one way ANOVA Field et al. [2012]	Repeated measures one way ANOVA (.0000)		② ④ ⑤ ⑥ ⑦ ⑨	✓
	Kruskal Wallis	(4.51825e-06)	② ④ ⑦ ⑨	—
	F-test	(1.24278e-07)	② ④ ⑤ ⑥ ⑦ ⑨	—
	Friedman	(5.23589e-11)	② ④ ⑦ ⑨	✓
	Factorial ANOVA	(1.24278e-07)	② ④ ⑤ ⑥ ⑨	✓
Two-way ANOVA Field et al. [2012]	Two-way ANOVA	(3.70282e-17)	② ④ ⑤ ⑨	—
	Bootstrap	(<0.05)		✓
Chi Square Field et al. [2012]	Chi Square	(4.76743e-07)	② ④ ⑨	✓
	Fisher's Exact	(4.76743e-07)	② ④ ⑨	✓

*① one variable, ② two variables, ③ two or more variables, ④ continuous vs. categorical vs. ordinal data, ⑤ normality, ⑥ equal variance, ⑦ dependent vs. independent observations, ⑧ exactly two groups, ⑨ two or more groups

We sampled 12 data sets and examples from R tutorials (Kabacoff [2011] and Field et al. [2012]). These included eight parametric tests, four non-parametric tests, and one Chi-square test. We chose these tutorials because they appeared in two of the top 20 statistical textbooks on Amazon and had publicly available data sets, which did not require extensive data wrangling.

We translated all analyses into Tea and encoded any assumptions explicitly stated in the tutorial. Tea selected tests based only on the data and the assumptions expressed in the Tea program. Where Tea disagreed with the tutorials, either (1) the tutorial authors chose the wrong analyses or (2) the tutorial authors had implicit assumptions about the data that did not hold up to statistical testing.

For nine out of the 12 tutorials, Tea suggested the same statistical test (see Table 3.1). For three out of 12 tutorials, which used a parametric test, Tea suggested using a non-parametric alternative instead. Tea's recommendation of using a non-parametric test instead of a parametric one did not change the statistical significance of the result at the .05 level. Tea suggested non-parametric tests based on the Shapiro-Wilk test for normality. It is possible that tutorial authors visualized the data to make implicit assumptions about the data, but this practice or conclusion was not made explicit in the tutorials.

For the two-way ANOVA tutorial from Field et al. [2012], which studied how gender and drug usage of individuals affected their perception of attractiveness, a precondition of the two-way ANOVA is that the dependent measure is normally distributed in each category. This precondition was violated. As a result, Tea defaulted to bootstrapping the means for each group and reported the means and confidence intervals. For the pointbiserial correlation tutorial from Field et al. [2012], Tea also defaulted to bootstrap for two reasons. First, the precondition of normality is violated. Second, the data uses a dichotomous (nominal) variable, which invalidates Spearman's ρ and Kendall's τ.

Tea generally agrees with expert recommendations and is more conservative in the presence of non-normal data, minimizing the risk of false positive findings.

Does Tea avoid common mistakes made by non-expert users?

Our goal was to assess whether any of the tests suggested by Tea (i.e., valid candidate tests) or any of the invalid candidate tests would lead to a different conclusion than the one drawn in the tutorial. Table 3.1 shows the results. Specifically, emphasized p-values indicate instances for which the result of a test differs from the tutorial in terms of statistical significance at the .05 level.

For all of the 12 tutorials, Tea's suggested tests led to the same conclusion about statistical significance. For two out of the 12 tutorials, two or more candidate tests led to a different conclusion. These candidate tests were invalid due to violations of independence or normality.

To summarize, the evaluation shows us that (i) Tea can replicate and even improve upon expert choices and (ii) Tea could help novices avoid common mistakes and false conclusions.

3.4 Discussion: Key tensions

- inflated alpha - inherent tension in executing multiple statistical tests vs. sensitivity

3.5 Limitations, Ongoing work, Future directions

Great for a class of relatively simple hypotheses and research questions, but the kinds of questions that analysts want to ask about their domain using data are more complex that simple hypotheses.

Testing to estimation

NHST to statistical modeling

How do we support this larger and more complex class of use cases? Can we generalize our approach in Tea–making implicit assumptions explicit and automatically reasoning about these assumptions to identify valid analyses–to statistical models? To answer this question, we set out to develop a holistic understanding of how analysts translate research questions into statistical analyses.

3.5.1 Ongoing development

With teams of undergrads, I have continued to improve Tea in two specific ways.

First, recent development has focused on updating the outputs of Tea to include (i) interactive visualizations and (ii) text for reporting the statistical results in the American Psychological Association's recommended formats for each valid statistical test. The interactive visualizations aim to illustrate what the results of the statistical tests mean, such as scatterplots for correlations and heatmaps for the Chi-squared test. We selected the visualizations for each test based on recommendations from Franconeri et al. **?**, what existing tools such as JMP SAS [2020] already use, and our own experiences using and trying to communicate statistical results. Development and initial user testing is on track to wrap up by the end of spring quarter.

Second, a usability issue with Tea's current API is its reliance of "magic strings." We are currently refactoring the API to be more object-oriented by extending Tisane's variables data classes. We hope this revision will be more usable with "free" help from existing IDEs such as VSCode that provide API suggestions inline when specifying parameters.

Both features will be incorporated into a new release of Tea, which I have currently scheduled for June, 2022.

3.6 Summary of Contributions

A common approach to assessing support for conceptual hypotheses in data is to use statistical tests (e.g., Student's t-test, Chi-Square test, ANOVA). Statistical testing requires analysts to grapple with their conceptual hypotheses, know a number of tests and when they are applicable (i.e., know the preconditions for when these tests hold), assess the applicability of tests (i.e., check preconditions), and pick and implement specific tests using low-level APIs.

Tea's key insight is that we can reformulate statistical test selection as a constraint satisfaction problem. We designed and implemented a higher-level DSL around this insight that takes an analyst's hypothesis and assumptions about their data as input and provides the results of executing valid statistical tests as output. In an evaluation, we found that Tea avoids faulty test selection and conclusions that are easy to make using existing tools.

Tea demonstrates the feasibility and benefit of developing systems that emphasize *higher-level abstractions* and *automated reasoning* for statistical tests (Section 1). However, Using statistics to answer real-world questions requires going beyond statistical testing to grappling with statistical modeling and effect estimation. Next, we consider how our approach generalizes to a larger class of statistical analyses.

This work was done in collaboration with Maureen Daum, Jared Roesch, Sarah E. Chasins, Emery Berger, René Just, and Katharina Reinecke. It was originally published and presented at ACM UIST 2019 cite. Since publication, multiple people, including most notably Shreyash Nigam, Reiden Chea, and Annie Denton, have contributed to updating and improving Tea.

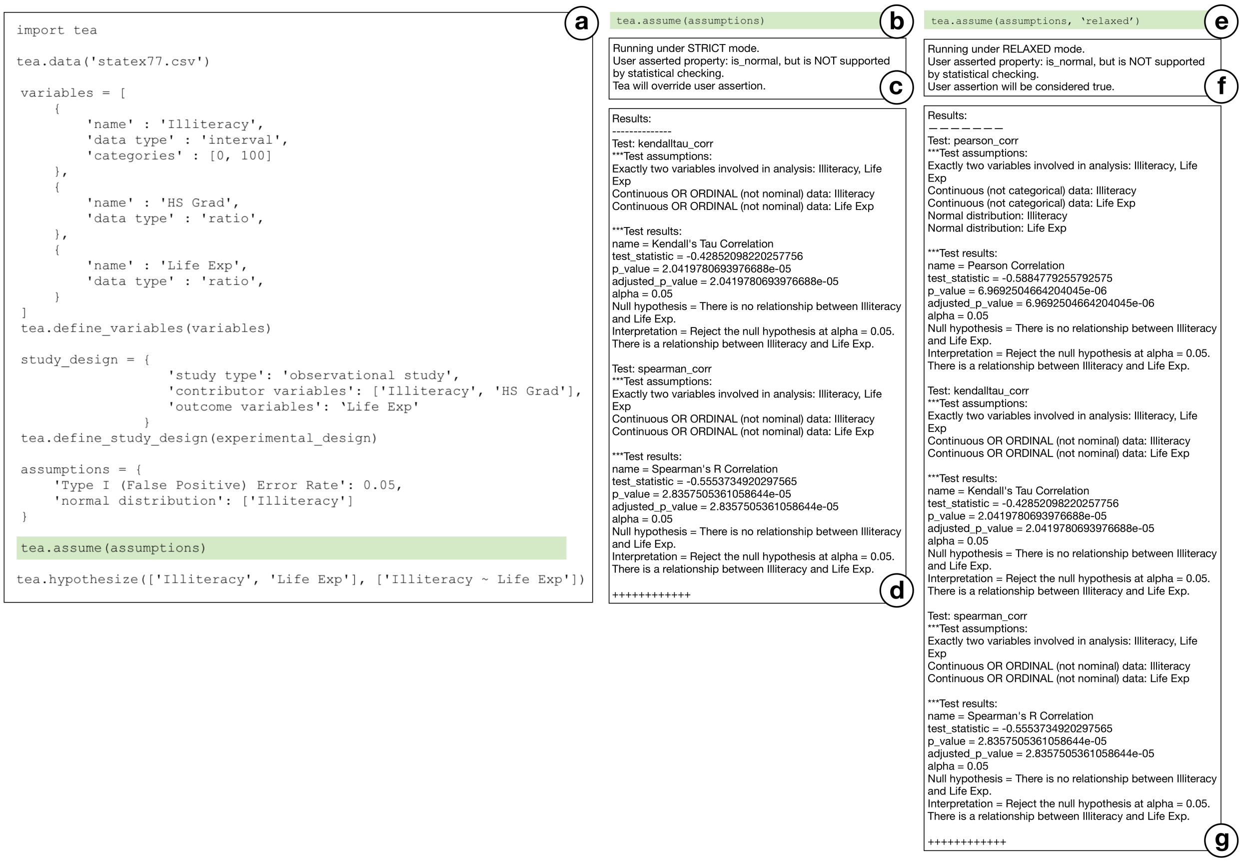


Figure 3.2: Tea program and its mode-dependent executions. a) Tea program that aims to determine if two contributor variables, ‘Illiteracy‘ and ‘HS Grad’ that may predict a third outcome variable ‘Life Exp’, are correlated. The user asserts that ‘Illiteracy’ is normally distributed. b) By default, Tea executes programs in the *strict* mode. c) Warning that Tea disagrees with the user and will override the user’s assertion that ‘Illiteracy’ is normally distributed in the *strict* mode. d) Results without the parametric test since Tea overrides user’s assertion. e) A single line change can modify Tea to execute a program in *relaxed* mode. f) Warning that Tea cannot verify normality for ‘Illiteracy’ but will defer to user’s assertion. g) Results with the parametric test since Tea proceeds as if ‘Illiteracy’ was normally distributed.

Chapter 4

Hypothesis Formalization: A conceptual framework describing how analysts translate research questions into statistical analyses

Rephrase the language describing code and data releases.

Consider a census researcher who asks, "In the United States (U.S.), how does an individual's sex relate to their annual income?" Drawing upon their prior experiences and exploratory data visualizations, the researcher knows that income in the U.S. is skewed, and they want to know how the distributions of income among males and females differ (step i). However, before implementing, they (implicitly) define their causal model: The researcher knows that other factors, such as education and race, may be associated with employment opportunities, which may then influence income. As such, they refine their conceptual hypothesis–that sex influences income–to consider the possible effects of race, education, sex, and their interactions on income. They plan to fit a generalized linear model with race, education, sex, and their two-way interactions as predictors of income (step ii). They start implementing a script to load and model data (step iii). The researcher receives a small table of results and is surprised to receive a convergence warning. After further investigation, they simplify their model and remove the interaction effects to see how that may affect convergence (revise step iii). This time, their model's inference algorithm converges, and they interpret the results (step iv), but they really want to study how sex and race interact, so they return to implementation (step iii) and proceed as before, iteratively removing and adding effects and changing computational parameters, and as a by-product shifting which high-level conceptual hypothesis is reflected in the model.

Performing statistical data analysis goes well beyond invoking the correct statistical functions in a software library. As seen with the census researcher, statistical analyses require(i) translating high-level, domain-specific questions and hypotheses into specific statistical questions Carver et al. [2016]; (ii) identifying statistical models to answer the statistical questions; (iii) implementing and executing these statistical models, typically with the help of software tools; and (iv) interpreting the results, considering the domain-specific questions and applying analytical reasoning. Analysts must go back and forth between conceptual hypothesis and model implementation realities, grappling with domain knowledge, limitations of data, and statistical methods.

We refer to the process of translating a conceptual hypothesis into a computable statistical model as *hypothesis formalization*. This process is messy and under-scrutinized in prior work. Consequently, we investigate the steps, considerations, challenges, and tools involved. Based on our findings, we define hypothesis formalization as a dual-search process Klahr and Dunbar [1988] that involves developing and integrating cognitive representations from two different perspectives—conceptual hypotheses and concrete model implementations. Analysts move back and forth between these two perspectives during formalization while balancing conceptual, data-driven, statistical, and implementation constraints. Figure **??** summarizes our definition and findings. Specifically, this chapter addresses the following questions to develop our definition of hypothesis formalization:

- **RQ1 - Steps**: What is the range of steps an analyst might consider when formalizing a hypothesis? How do these steps compare to ones that we might expect based on prior work?
- **RQ2 - Process**: How do analysts think about and perform the steps to translate their hypotheses into model implementations? What challenges do they face during this process?
- **RQ3 - Tools**: How might current software tools influence hypothesis formalization?

To sensitive ourselves to the steps (**RQ1 - Steps**) and considerations (**RQ2 - Process**) involved in hypothesis formalization, we compared and contrasted existing models and descriptions of data analysis in prior work. We augmented our deep dive into prior work with a formative content analysis of 50 randomly sampled research papers from five different venues, including Psychological Science and Nature. We find that researchers decompose their research hypotheses into specific sub-hypotheses, derive proxy variables from theory and available data, and adapt statistical analyses to account for data collection procedures. A key takeaway from prior work and the formative content analysis was the "hypothesis refinement loop" in Figure **??**.

To validate and deepen our understanding of hypothesis formalization (**RQ1 - Steps** and **RQ2 - Process**), we designed and conducted a lab study in which we observed 24 analysts develop and formalize hypotheses in-situ. We find that analysts foreground implementation concerns, even when brainstorming hypotheses, and try to fit their hypotheses and analyses to prior experiences and familiar tools, suggesting a strong influence of tools (**RQ3 - Tools**). Thus, the lab study reinforced the hypothesis refinement loop, surfaced the "model implementation loop," and raised questions about the role of tools.

To identify how tools may shape hypothesis formalization (**RQ3 - Tools**), we reviewed 20 statistical software tools. We find that although the tools support nuanced model implementations, their low-level abstractions can focus analysts on statistical and computational details at the expense of higher-level reasoning about initial hypotheses. Tools also do not aid analysts in identifying reasonable model implementations that would test their conceptual hypotheses, which may explain why analysts in our lab study relied on familiar approaches, even if sub-optimal. Furthermore, our tools review confirmed that the dual processes inform one another during hypothesis formalization.

Taken together, our findings help us define the hypothesis formalization framework, as summarized in Figure **??**, and suggest **three design implications** for tools to more directly support hypothesis formalization: (i) show the relationships between related statistical models that seem syntactically different from each other, (ii) provide higher-level abstractions for expressing conceptual hypotheses and partial model specifications, and (iii) develop bidirectional computational assistance for authoring causal models and relating them to statistical models.

By defining and characterizing hypothesis formalization, we situate data analysis in a larger model of scientific discovery, identify specific problem solving strategies used in hypothesis formalization that demonstrate how data analysis (and science) is a practice, and identify opportunities for future software to improve the transparency and reproducibility of analyses by explicitly supporting pathways and loops through hypothesis formalization.

4.1 Background and Related Work

To do (**??**) Our work integrates and builds upon prior research on frameworks of scientific discovery, theories of sensemaking, statistical practices, and empirical studies of data analysts.

4.1.1 Statistical Thinking

Statistical thinking and practice require differentiating between *domain* and *statistical* questions. The American Statistical Association (ASA), a professional body representing statisticians, recommends that universities teach this fundamental principle in introductory courses (see Goal 2 in Carver et al. [2016]).

Similarly, researchers Wild and Pfannkuch emphasize the importance of differentiating between and integrating statistical knowledge and context (or domain) knowledge when thinking statistically Pfannkuch [1997]; Pfannkuch et al. [2000]; Wild and Pfannkuch [1999]. They propose a four step model for operationalizing ideas ("inklings") into plans for collecting data, which are eventually statistically analyzed. In their model, analysts must transform "inklings" into broad questions and then into precise questions that are then finally turned into a plan for data collection (see Figure 2 in Wild and Pfannkuch [1999]). Statistical and domain knowledge inform all four stages. However, it is unknown what kinds of statistical and domain knowledge are helpful, how they are used and weighed against each other, and when certain kinds of knowledge are helpful to operationalize inklings. Our work provides more granular insight into Wild and Pfannkuch's proposed model of operationalization and aims to answer when, how, and what kinds of statistical and domain knowledge are used during statistical data analysis.

More recently, in *Statistical Rethinking* McElreath [2020], McElreath proposes that there are three key representational phases involved in data analysis: conceptual hypotheses, causal models underlying hypotheses (which McElreath calls "process models"), and statistical models. McElreath, like the ASA and Wild and Pfannkuch, separates domain and statistical ideas and discusses the use of causal models as an intermediate representation to connect the two. McElreath emphasizes that conceptual hypotheses may correspond to multiple causal and statistical models, and that the same statistical model may provide evidence for multiple, even contradictory, causal models and hypotheses. McElreath's framework does not directly address how analysts navigate these relationships or how computation plays a role, both of which we take up in this paper.

Overall, our work provides empirical evidence for prior frameworks but also (i) provides more granular insight into *how* and *why* transitions between representations occur and (ii) scrutinizes the role of *software and computation* through close observation of analyst workflows in the lab as well as through a follow-up analysis of statistical software. Based on these observations, we also speculate on how tools might better support hypothesis formalization.

4.1.2 Dual-search Model of Scientific Discovery

4.2 Statistical data analysis as part of scientific discovery

Grolemund and Wickham's depiction of the analysis process parallels Klahr and Simon's framework of scientific discovery. Klahr and Simon characterized scientific discovery as a dual-search process involving the development and evaluation of hypotheses and experiments Klahr and Dunbar [1988]. They posited that scientific discovery involved tasks specific to hypotheses (e.g., revising hypotheses) and to experiments (e.g., analyzing data collected from experiments), which they separated into two different "spaces," and tasks moving between them, which is where we place hypothesis formalization. Extending Klahr and Simon's two-space model, Schunn and Klahr proposed a more granular four-space model involving data representation, hypothesis, paradigm, and experiment spaces Schunn and Klahr [1995, 1996]. In the four-space model, conceptual hypothesizing still lies in the hypothesis space, and hypothesis testing and statistical modeling lies in the paradigm space. As such, hypothesis formalization is a process connecting the hypothesis and paradigm spaces. In Schunn and Klahr's four-space model, information flows unidirectionally from the hypothesis space to the paradigm space. In **??** we extend this prior research with evidence that the path from hypothesis and paradigm spaces is actually bidirectional. (see Figure **??**).

Klahr and Simon characterized scientific discovery as a dual-search process involving the development and evaluation of hypotheses and experiments Klahr and Dunbar [1988]. They posited that scientific discovery involved tasks specific to hypotheses (e.g., revising hypotheses) and to experiments (e.g., analyzing data collected from experiments), which they separated into two different "spaces," and tasks moving between them, which is where we place hypothesis formalization.

Extending Klahr and Simon's two-space model, Schunn and Klahr proposed a more granular four-space model involving data representation, hypothesis, paradigm, and experiment spaces Schunn and Klahr [1995, 1996]. In the four-space model, conceptual hypothesizing still lies in the hypothesis space, and hypothesis testing and statistical modeling lies in the paradigm space. As such, hypothesis formalization is a process connecting the hypothesis and paradigm spaces. In Schunn and Klahr's four-space model, information flows unidirectionally from the hypothesis space to the paradigm space. Here we extend this prior research with evidence that hypothesis formalization involves both concept-to-implementation and implementation-to-concept processes. (see Figure **??**). Figure 4.1 augments Schunn and Klahr's original diagram (Figure 1 in Schunn and Klahr [1995]) with annotations depicting how our content analysis of research papers and lab study triangulate a tighter dual-space search between hypothesis and paradigm spaces with a focus on hypothesis formalization. Our mixed-methods approach follows the precedent and recommendations of Klahr and Simon's Klahr and Simon [1999] study of scientific discovery activities.

4.2.1 Theories of Sensemaking

Human beings engage in *sensemaking* to acquire new knowledge. Several theories of sensemaking Pirolli and Card [2005]; Russell et al. [1993]; Klein et al. [2007] describe how and when human beings seek and integrate new data (e.g., observations, experiences, etc.) to develop their mental models about the world.

Russell et al. Russell et al. [1993] emphasize the importance of building up and evaluating external representations of mental models, and define sensemaking as "the process of searching for a representation and encoding data in that representation to answer task-specific questions." External representations are critical because they influence the quality of conclusions reached at the end of

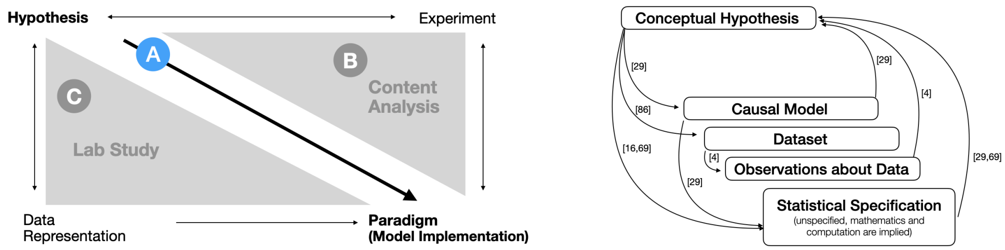


Figure 4.1: Relationship between hypothesis formalization and prior work. *Left:* Schunn and Klahr's four-space model of scientific discovery (stylized adaptation from Figure 1 in Schunn and Klahr [1995]), which includes unidirectional information flow from the hypothesis space to the paradigm space (which includes model implementation). Hypothesis formalization (A) is focused on a tighter integration and the information flow between hypothesis and paradigm spaces. Specifically, the information flow is bidirectional in hypothesis formalization. Our content analysis (B) and lab study (C) triangulate the four-space model to understand hypothesis formalization from complementary perspectives. *Right:* Hypothesis formalization steps also identified in prior work on theories of sensemaking, statistical thinking, and data analysis workflows (citations included to the right of the arrows). Hypothesis formalization is finer grained and involves more iterations. While prior work broadly refers to mathematical equations, partial model specifications, and computationally tuned model implementations as statistical specifications, hypothesis formalization differentiates them. As a whole, this paper provides empirical evidence for theorized loops between conceptual hypothesis and statistical specification (see Figure **??**).

the sensemaking process and affect how much time and effort is required in the process. Some representations may lead to insights more quickly. Russell et al. describe the iterative process of searching for and refining external representations in a "learning loop complex" that involves transitioning back and forth between (i) searching for and (ii) instantiating representations.

Grolemund and Wickham argued for statistical data analysis as a sensemaking activity Grolemund and Wickham [2014]. They emphasize the (1) bidirectional nature of updating mental models of the world and hypotheses based on data and collecting data based on hypotheses and (2) the process of identifying and reconciling discrepancies between hypotheses and data. Their depiction of the analysis process parallels Klahr and Simon's framework of scientific discovery.

and proposed a theory of data analysis that includes a back and forth between an analyst's "schema" of how a phenomenon occurs in the world, a statistical model, and data. Similar to Russell et al., Grolemund and Wickham's model demonstrates the importance of representing and re-representing conceptual knowledge in schema and statistical models that are updated with more data. Analysts' domain expertise influence their schemas, which represent conceptual knowledge about known and unknown causal mechanisms, for example. Analysts' conceptual schema directly inform their hypotheses, which are statistical predictions represented in statistical models. These statistical models are then compared to collected data, and any discrepancies between the data and hypothesis require analysts to re-examine and possibly update their statistical model, schema, or both.

In this paper, we consider hypothesis formalization to be a learning loop Russell et al. [1993]

where the conceptual hypothesis is an external representation of a set of assumptions analysts may have about the world (e.g., an implicit causal model), that ultimately affects which models are specified and which results are obtained. We found that that there are smaller learning loops as analysts search for and revise intermediate representations, such as explicit causal models, mathematical equations, or partially specified models. The hypothesis and model refinement loops can themselves be smaller learning loops embedded in the larger loop of hypothesis formalization.

Extending Grolemund and Wickham's model, our work on hypothesis formalization differentiates between conceptual and statistical hypotheses and probes the phases an analyst must go through to encode a conceptual hypothesis into a statistical model.

In summary, our work differs in (i) scope and (ii) method from prior work in HCI on data analysis practices. Whereas hypothesis formalization has remained implicit in prior descriptions of data analysis, we explicate this specific process. While previous researchers have relied primarily on post-analysis interviews with analysts, our lab study (Section 4.4) enables us to observe decision making during hypothesis formalization in-situ.

4.3 Formative content analysis

To complement our in-depth synthesis of prior work, we conducted a formative content analysis of 50 peer-reviewed publications from five different domains.

Methods

We randomly sampled ten papers published in 2019 from each of the following venues: (1) the Proceedings of the National Academy of Sciences (PNAS), (2) Nature, (3) Psychological Science (PS), (4) Journal of Financial Economics (JFE), and (5) the ACM Conference on Human Factors in Computing Systems (CHI). We sampled papers that used statistical analyses as either primary or secondary methodologies. Our sample represents a plurality of domains and recent practices. [1]

The first two authors iteratively developed a codebook to code papers at the paragraph-level. The codebook contained five broad categories: (i) research goals, (ii) data sample information, (iii) statistical analysis, (iv) results reporting, and (v) computation. Each category had more specific codes to capture more nuanced differences between papers. This tiered coding scheme enabled us to see general content patterns across papers and nuanced steps within papers. The first two authors reached substantial agreement (IRR = .69 - .72) even before resolving disagreements. The first

[1] Google Scholar listed the venues among the top three in their respective areas in 2018. Venues were often clustered in the rankings without an obvious top-one, so we chose among the top three based on ease of access to publications (e.g., open access or access through our institution). Some papers were accepted and published before 2019, but the journals had included them in 2019 issues.

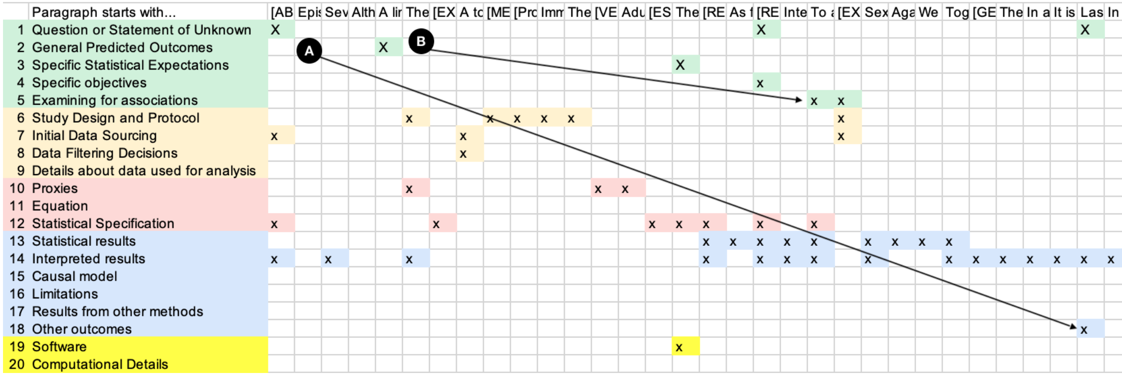


Figure 4.2: Formative content analysis: example reorderable matrix for ?.
We visualized each paper in our sample as a "reorderable matrix" Bertin [2011] to aid in detecting patterns in papers' structure and content that could indicate how researchers formalized their hypotheses. The rows represent the codes in our codebook, colored according to the five broad categories of codes: research goals (rows 1-5, green), sample information (rows 6-9, orange), statistical analysis details (rows 10-12, red), reporting of results (rows 13-18, blue), and computational details (rows 19-20, bright yellow). The columns are the paragraphs, which are indexed by their first sentences, ordered left to right. In a paragraph's column, there is an "X" for each code the paragraph received. Paragraphs have multiple codes if they contain multiple types of information. Among the ten visual patterns we noticed across our sample and subsequently looked for in each paper, two stand out in this paper. (A) As the paper progresses (visually moving left to right), the paper's focus shifts from research goals to sample information to statistical analysis to results, as indicated by the arrow labeled A. Largely expected, this pattern helps to validate our coding method. Also, there is only one paragraph that discusses statistical software. (B) Researchers discuss research goals and questions throughout the paper. Interestingly, in the middle of the paper, when the researchers discuss their goals in greater detail, the researchers discuss them in increasing specificity, as indicated by the arrow labeled B. We were able to detect this pattern across papers by iterating on how to order the research goal codes (rows 1-5, green). The final order lists codes in increasing specificity from top (row 1) to bottom (row 5). Pattern B suggests that researchers refine their hypotheses during hypothesis formalization, which may involve specifying proxies and statistical methods. Our supplementary materials discuss additional patterns in this paper and across our entire sample.

three authors then (i) read and coded all sections of papers except the figures, tables, and auxiliary materials that did not pertain to methodology[2]; (ii) discussed and summarized the papers' goals and main findings to ensure comprehension and identify contribution types; and (iii) visualized each paper as a "reorderable matrix" Bertin [2011].

We adapted Bertin's "reorderable matrix" Bertin [2011], an interactive visualization technique for data exploration, in our analysis. We visualized each paper in our sample as a matrix where each row represented a code in our codebook and each column represented a coded paragraph. We fixed the order of paragraphs to match the paper's progression. We colored codes (rows) according to their categories in our codebook, repeatedly reordered the rows representing codes, and transposed the matrices to detect visual patterns in the papers. Figure 4.2 shows an example matrix.

[2]PNAS and Nature papers included a materials and methods section after references that were distinct from extended tables, figures, and other auxiliary material. We coded the materials and methods sections in the appendices and included them in the content analysis. Our supplementary material describes our process in greater detail.

The visual representation of papers' content and structure helped us notice common patterns across papers and guided our follow-up analyses and discussions about what steps (**RQ1 - Steps**) and considerations (**RQ2 - Process**) researchers reported having during hypothesis formalization. Across multiple papers, the matrices showed how researchers typically start with broader research goals that they decompose into specific hypotheses (i.e., hypothesis refinement) over the course of a paper section, for example. Within a single paper, the matrices visually showed patterns of how researchers motivated and pieced together multiple experiments and interpreted statistical results in order to make a primary scientific argument. Our supplementary materials include our codebook with definitions and examples as well as a summary, citation, and annotated matrix for each paper.

Findings

The content analysis confirmed prior findings on (i) the connection between hypotheses and causal models (e.g.,McElreath [2020]), (ii) the importance of proxies to quantify concepts, and (iii) the constraints that data collection design and logistics place on modeling. Extending prior work, the content analysis also (i) suggested that decomposing hypotheses into specific objectives is a mechanism by which conceptual hypotheses relate to causal models; (ii) crystallized the hypothesis refinement loop involving conceptual hypotheses, causal models and proxies; and (iii) surfaced the dual-search nature of hypothesis formalization by suggesting that model implementation may shape data collection.

Limitations

The major limitation of analyzing published papers is the disconnect between actual and reported analytical practice. The pressures to write compelling scientific narratives Kerr [1998] likely influence which aspects of hypothesis formalization are described or omitted. For instance, in practice, model implementations may constrain data collection more often than we found in our sample. Nevertheless, the lack of information in prior work and the content analysis suggests that hypothesis formalization remains an opaque process deserving of greater scrutiny. Hypothesis formalization may explain how analysts determine which tools to use and how domain expertise may influence the analytical conclusions reached.

4.3.1 Expected Steps in Hypothesis Formalization

Towards our first two research questions about what actions analysts take to formalize hypotheses (**RQ1 - Steps**) and why (**RQ2 - Process**), prior work and our formative content analysis suggest that hypothesis formalization involves steps in three categories: conceptual, data-based, and statistical. *Conceptually,* analysts develop conceptual hypotheses and causal models about their domain that guide their data analysis. With respect to *data*, analysts explore data and incorporate insights from exploration, which can be top-down or bottom-up, into their process of formalizing hypotheses. The *statistical* concerns analysts must address involve mathematical and computational concerns, such as identifying a statistical approach (e.g., linear modeling), representing the problem mathematically (e.g., writing out a linear model equation), and then implementing those using software. In our work, we find evidence to support separating statistical considerations into concerns about mathematics, statistical specification in tools, and model implementation using tools.

A key observation about prior work is that there is a tension between iterative and linear workflows during hypothesis formalization. Although sensemaking processes involve iteration, concerns about methodological soundness, as evidenced in pre-registration efforts that require researchers to specify and follow their steps without deviation, advocate for, or even impose, more linear processes. More specifically, theories of sensemaking that draw on cognitive science, in particular Russell et al. [1993]; Grolemund and Wickham [2014], propose larger iteration loops between conceptual and statistical considerations. Some textbooks and research concerning statistical thinking and practices Wild and Pfannkuch [1999]; Carver et al. [2016] appear less committed to iteration while other researchers and practitioners in applied statistics emphasize *workflows* for iterating on statistical models **???**. Workflows (e.g., model expansion) can help researchers start with simple models and build up to more complex ones by incrementally testing and refining their understanding of characterstics of the data, the model fitting algorithms, and computational settings **??**Gabry et al. [2019]. Moreover, empirical work in HCI on data analysis embraces iteration during exploration and observes iteration during some phases of confirmatory data analysis, such as statistical model choice, but not in others, such as tool selection. In our work, we are sensitive to this tension and aim to provide more granular insight into iterations and linear processes involved in hypothesis formalization. We also anticipate that the steps identified in prior work will recur in our lab study, but we do not limit our investigation to these steps.

4.4 Exploratory Lab Study

To address the limitation of the content analysis, understand analysts' considerations (**RQ2 - Process**) while formalizing their hypotheses (**RQ1 - Steps**), and examine the role of statistical software in this process (**RQ3 - Tools**), we designed and conducted a virtual lab study with freelance data workers who approach the hypothesis formalization and analysis process with expectations of rigor but without the pressure of publication.

4.4.1 Methods

Data workers: We recruited 24 data workers with experience in domains ranging from marketing to physics to education through Upwork (22) and by word of mouth (2).[3]

Twelve data workers held occupations as scientists, freelance data scientists, project managers, or software engineers. Six were currently enrolled in or had just finished graduate programs that involved data analysis. Five identified as current or recent undergraduates looking for jobs in data science. One was an educator. Data workers self-reported having significant experience on a 10-point scale adapted from a scale for programming experience Feigenspan et al. [2012] (min=2, max=10, mean=6.4, std=2.04) and would presumably have familiarity with hypothesis formalization.

The lab study enables us to contrast normative expert practices (found in prior work and our formative content analysis) to observed practices with data workers who are not statistical experts but still work in real-world analysis settings (i.e., research, marketing, consulting). A benefit of studying these data workers is that they are likely to benefit most from new tools.

Protocol: We designed and conducted a lab study with three parts. Parts 1 and 3 were recorded and automatically transcribed using Zoom. We compensated data workers $45 for their time. The first author conducted the study and took notes throughout.

Part 1: Structured Tasks. Part 1 asked data workers to imagine they were leading a research team to answer the following research question: "What aspects of an individual's background and demographics are associated with income after they have graduated from high school?"[4] We asked data workers to complete the following tasks:

- *Task 1: Hypothesis generation.* Imagining they had access to any kind of data thinkable, data workers brainstormed at least three hypotheses related to the research question.
- *Task 2: Conceptual modeling.* Next, data workers saw a sample data schema and developed a conceptual model for one or more of their hypotheses. We used the term "conceptual model" instead of "causal model" to avoid (mis)leading data workers. We provided the following definition: "A conceptual model summarizes the process by which some outcome occurs. A conceptual model specifies the factors you think influence an outcome, what factors you think do not influence an outcome, and how those factors might interact to give rise to the outcome."
- *Task 3: Statistical model specification.* Finally, we presented data workers with a sample dataset and instructed them to specify but not implement a statistical model to test one or more of their hypotheses.

[3] We refer to our participants as data workers because they work with data but do not represent the entire population of data scientists, which may include statistical experts.

[4] We chose the open-ended research question about income after high school because we expected it to be widely approachable and require no domain expertise to understand.

After the three tasks, we conducted a semi-structured interview with data workers about (i) their validity concerns[5] and (ii) experiences. To help us contextualize our observations and assess the generalizability of our findings, we asked data workers to compare the study's structure and tasks to their day-to-day data analysis practices.

Part 2: Take-home analysis. After the first Zoom session, data workers implemented their analyses using the previously shown dataset, shared any analysis artifacts (e.g., scripts, output, visualizations, etc.), and completed a survey about their implementation experience. Prior to Part 3, the first author reviewed all submitted materials and developed participant-specific questions for the final interview.

Part 3: Final Interview. The first author asked data workers to give an overview of their analysis process and describe the hypotheses they tested, how their analysis impacted their conceptual model and understanding, why they made certain implementation choices, what challenges they faced (if any), and any additional concerns about validity.

Materials: The data schema and dataset used in the study came from a publicly available dataset from the Pew Research Center Suh [2014]. Each task was presented in a separate document. All study materials are included as supplementary material.

Analysis: The first author reviewed the data workers' artifacts multiple times to analyze their content and structure; thematically analyzed notes and transcripts from data workers' Zoom sessions; and regularly discussed observations with the other authors throughout analysis.

4.4.2 Findings and Discussion

Eighteen of the 24 data workers we recruited completed all three parts of the study. The other six data workers completed only the first Zoom session. In our analysis, we incorporate data from all data workers for as far as they completed the study.

We found that data workers had four major steps (**RQ1 - Steps**) and considerations (**RQ2 - Process**): (i) identifying or creating proxies, (ii) fitting their present analysis to familiar approaches, (iii) using their tools to specify models (**RQ3 - Tools**), and (iv) minimizing bias by relying on data. Data workers also faced challenges acquiring and incorporating domain and statistical knowledge (**RQ2 - Process**).

Data workers consider proxies and data collection while articulating hypotheses.

We encouraged data workers to not consider the feasibility of collecting data while brainstorming hypotheses. Yet, while brainstorming hypotheses, data workers expressed concern with how to

[5] If data workers were unfamiliar with the term "validity," we rephrased the questions to be about "soundness" or "reliability."

measure constructs [D2, D5, D8, D12, D18, D22, D24] and how to obtain data [D2, D6, D8, D9, D11, D21, D24].

For instance, D18, a computer science student who had worked on more than five data analysis projects, grappled with the idea of 'privilege' and how to best quantify it:

> "*I'm trying to highlight the fact that those who will be privileged before graduation...that experience will enable them to make again more money after graduation. I won't say 'privilege' because we need to quantify and qualify for that...it's just an abstract term.*"

Eventually, D18 wrote two separate hypotheses about 'privilege,' operationalizing it as parental income: (1) "People with higher incomes pre graduating, end up having higher differences between pre and post graduation incomes than those with lower incomes pre graduation." and (2) "People with parents with lower incomes tend to have lower incomes pre graduation than those with parents with higher incomes."

D18 continued to deliberate 'privilege' as measured by low and high income, saying, "*...again you need to be careful with low and high because these are just abstract terms. We need to quantify that. What does it mean to be 'low?' What does it mean to be 'high?'*". Finally, D18 decided to "*maybe use the American standards for low income and high income.*" Although an accepted "American standard" may not exist, D18 nevertheless believed that cultural context was necessary to specify because it could provide a normalizing scale to compare income during analysis, demonstrating how data workers plan ahead for statistical modeling while brainstorming and refining hypotheses.

Similarly, D2, a freelance data scientist, was very specific about how to measure personality: "More extraverted individuals (extraversion measured using the corresponding social network graph) are likely to achieve higher yearly income later in life."

In the presence of the data schema, more data workers were concerned with proxies [D2, D5, D6, D7, D8, D9, D16, D18, D21]. Some even adapted their working definitions to match the available data, similar to how researchers in the content analysis determined proxies based on data. For instance, D8, who hypothesized that "individuals interested in STEM fields tend to earn more post high school than individuals interested in other fields," operationalized "interest" as "Major" — a variable included in the data schema — even though they had previously brainstormed using other proxies such as club attendance in high school.

These data workers' closely related considerations of data and concept measurement demonstrate how conceptual hypotheses and data collection may inform each other, corroborating our findings from the content analysis.

Create new variables:

Adj_annual_income - take the midpoint of the ranges in the Annual Income column as a numeric value. (numeric)

State_avg_income - find the average income of individuals in each state from established benchmarks. (numeric)

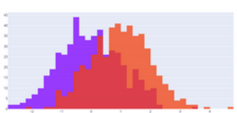
Income_over_avg - take the difference between each individual's income with the average for their state.

Testing Major vs income: take all rows with a college degree (2 year associate and up) & major. Omit rows with no info on income.

For each major, calculate the average *Adj_annual_income*.

Also, calculate the average Adj_annual_income for all the college rows from above.

Create a set of histograms (one for each major) showing the spread of Adj_annual_income for the people in that group. The histograms should share the same x axis. The bins will be normalized to sum to 100% for each major group.



Arrange the data like so

Major	Avg Income (within major)	Avg income (sample population)
Bio	####	####
Stats	####	####
etc.	####	####

Chi-squared test.

H_0: for each major group, the average income is equal to the entire sample population's average income. That is, no single group has a significant difference in avg income from the sample population.

H_A: at least one of the major groups has an average income that's significantly different from the sample population.

Test for a p-value <= 0.05

One caveat of our selected test is even if we are able to reject H_0, we can't make conclusions about which major group is the one making the different. It's possible that just one group is; it's possible that *every* group is significantly different from the population writ large.

Figure 4.3: Sample statistical specification (D8).

The lab study tasked analysts to specify their statistical models without considering implementation. We expected analysts would represent their statistical models using statistical test names or mathematical equations. Instead, most analysts specified statistical procedures for performing statistical models using todo lists and summaries of steps, which sometimes included mentions of software tools, showing that implementation was an important consideration and that tool familiarity may limit which statistical models analysts consider and implement. Data worker D8 specified their model through a combination of statistical test names (e.g., Chi-squared test) and a list (split across two pages) of detailed steps involved in creating new variables, cleaning and wrangling data, visualizing data, and testing their hypothesis.

Data workers consider implementation and tools when specifying statistical models.

When we asked data workers to specify their models without considering implementation, we anticipated they would name specific statistical tests (e.g., "ANOVA"), approaches (e.g., "linear regression" or "decision trees"), or write mathematical models (e.g., $Y = B_0 + B_1X_{age} + B_2X_{gender}$) that they could then implement using their tools because (a) some researchers in the literature survey did so in their papers and (b) several data workers mentioned having years of analysis experience. However, despite the explicit instruction to disregard implementation, 16 data workers provided to-do lists or summaries of steps to perform a statistical analysis as their model specifications [D1, D2, D3, D5, D7, D8, D9, D11, D12, D14, D16, D18, D20, D21, D22, D23, D24]. Of these 16 data workers, eight also named specific statistical tests in their descriptions [D3, D7, D8, D11, D12, D14, D18, D20].

For example, D8, a data science consultant with 7/10 analysis experience, specified a list of steps that included creating new variables that aggregated columns in the dataset, cleaning and wrangling the data, visualizing histograms, performing chi-squared test, and interpreting the statistical results. Notably, D8 also specified null and alternative hypotheses, which acted as an intermediate artifact during hypothesis formalization. Figure 4.3 shows D8's statistical specification.

Only four data workers named specific statistical methods without describing their steps [D4, D6, D15, D17]. Two data workers, D22, a neuroscientist by training with 8/10 analysis experience, and D19, an educator with 6/10 analysis experience, attempted to specify their models mathematically. D22 used the familiar R syntax: "Current Income ~ Educational attainment + Gender + Interactions of those two." On the other hand, D19 gave up because although they knew the general form of logistic regression, they did not know how to represent the specific variables in the model they wanted to perform.

The implementation and software details data workers discussed and included in their specifications suggest that data workers prefer to skip over mathematical equations and jump to specification and implementation in their tools. Although it is possible that study instructions primed data workers to respond about how they would perform, rather than represent, the task even after researcher clarifications, this would not explain the level of implementation detail data workers included. Nine data workers went so far as to mention specific libraries, even functions, that they would use to program their analyses [D3, D9, D12, D13, D14, D16, D19, D21, D23]. In their reflective interviews, data workers also expressed that they often do not specify models outside of implementing them, which D19 succinctly described:

> "*I don't normally write this down because all of this is in a [software] library.*"

Data workers' statistical knowledge appears to be situated in the programs they write, and their knowledge of and familiarity with tools constrains the statistical methods they explore and consider.

As such, tools may be a key point of intervention for guiding data workers toward statistical methods that may be unfamiliar but are best suited for their conceptual hypotheses.

Data workers try to fit analyses to previous projects and familiar approaches.

Data workers spent significant thought and time categorizing their analyses as "prediction," "classification," or "correlation" problems [D2, D3, D7, D10, D11, D18, D19, D21, D22]. To categorize, data workers relied on their previous projects. While reflecting on their typical analysis process, D21, a software engineer working in healthcare, said (emphasis added),

> "*I usually tend to jump...to look at data and **match [the analysis problem] with similar patterns** I have seen in the past and start implementing that or do some rough diagrams [for thinking about parameters, data type, and implementation] on paper...and start implementing it.*"

Data workers also looked at variable data types (i.e., categorical or continuous) to categorize. For example, D3, a freelance analyst, pivoted from thinking about **predicting** income to **classifying** income groups (emphasis added) based on data type information:

> "*The income, the column, the target value here, is categorical. I think maybe it wouldn't be a bad idea to see what **classification** tasks, what we could do. So instead of trying to **predict** because we're not trying to **predict an exact number**, it seems...like more of a **classification** problem...*"

A provocative case of adhering to prior experiences was D6, a psychological research scientist. Although several data workers were surprised and frustrated that income was ordinal in the dataset with categories such as"Under $10K," "$10K to $20K," "$20K to $30K," up to "150K+", none went so far as D6 to synthetically generate normally distributed income data so that they could implement the linear regression models they had specified despite saying they knew that income was not normally distributed.

When asked further about the importance of normal data, D6 described how they plan analyses based on having normal data, strive to collect normally distributed, and rely on domain knowledge to transform the data to be normal when it may not be after collection:

> "*...I feel like having non normal data is something that's like hard for us to deal with. Like it just kind of messes everything up like. And I know, I know it's not always assumption of all the tasks, but just that we tend to try really hard to get our variables to be normally distributed. So, you know, we might like transform it or, you know, kind of clean it like clean outliers, maybe transform if needed...I mean, it makes sense because*

> *like a lot of measures we do use are like depressive symptoms or anxiety symptoms and kind of they're naturally normally distributed...I can probably count on my hand the number of non parametric tests I've like included in manuscripts.*"

D6's description of their day-to-day analyses exemplifies the dual-search nature of hypothesis formalization: Data workers (i) jump from hypothesis refinement to model specification or implementation with specific proxies in mind and then (ii) collect and manipulate their data to fit their model choices.

We recognize that data workers may have taken shortcuts for the study they would not typically make in real life. Nevertheless, the constraints we imposed by using a real-world dataset are to be expected in real-world analyses. Therefore, our observations still suggest that rather than consider the nature and structure of their hypotheses and data to inform using new statistical approaches, which statistical pedagogy and theory may suggest, data workers may choose familiar statistical approaches and mold their new analyses after previous ones.

Data workers try to minimize their biases by focusing on data.

Throughout the study, data workers expressed concern that they were biasing the analysis process. Data workers drew upon their personal experiences to develop hypotheses [D5, D10, D13, D15, D16, D20, D21, D24] and conceptual models [D8, D12, D20, D24]. D12, a data analysis project manager, described how their personal experiences may subconsciously bias their investigation by comparing a hypothetical physicist and social worker answering the same research question:

> "*Whereas a social worker by design...they're meant to look at the humanity behind the numbers [unlike a physicist]. So like, they may actually end up with different results...actually sitting in front of this data, trying to model it.*"

A few data workers even refused to specify conceptual models for fear of biasing the statistical analyses [D10, D11, D19]. On the surface, data workers resisted because they believed that some relationships, such as the effect of age on income, were too "obvious" and did not warrant documentation [D10, D11]. However, relationships between variables that were "obvious" to some data workers were not to others. For instance, D10, a business analyst, described how income would plateau with age, but other data workers, such as D18, assumed income would monotonically increase with age.

When we probed further into why D10, D11, and D19 rejected a priori conceptual models, they echoed D10's belief that conceptual models "put blinders on you." Even the data workers who created conceptual models echoed similar concerns of wanting to "[l]et the model do the talking" in their implementations [D3, D15, D18, D19]. Instead of conceptual modeling, D10 chose to look at all n-ary relationships in the dataset to determine which variables to keep in a final statistical model, saying,

> "*It's so easy to run individual tests...You can run hypothesis tests faster than you can actually think of what the hypothesis might be so there's no need to really presuppose what relationships might exist [in a conceptual model].*"

Of course, one could start from the same premise that statistical tests are so easy to execute and conclude that conceptual modeling is all the more important to prioritize analyses and prevent false discoveries.

Similarly, data workers were split on whether they focused their implementation exclusively on their hypotheses or examined other relationships in the dataset opportunistically. Nine data workers stuck strictly to testing their hypotheses [D1, D4, D5, D6, D7, D11, D13, D20, D24]. However, five data workers were more focused on exploring relationships in the dataset and pushed their hypotheses aside [D2, D3, D10, D16, D18], and an additional four data workers explored relationships among variables not previously specified in their hypotheses in addition to their hypotheses [D14, D15, D17, D21]. D18 justified their choice to ignore their hypotheses and focus on emergent relationships in the data by saying that they wanted to be "*open minded based on the data...open to possibilities.*"

Data workers' concerns about bias and choice of which relationships to analyze (hypothesis only vs. opportunistic) highlight the tension between the two searches involved in hypothesis formalization: concept-first model implementations and implementation-first conceptual understanding. Conceptual models are intermediate artifacts that could reconcile the two search processes and challenge data workers' ideas of what "data-driven" means. However, given some data workers' resistance to prior conceptual modeling, workflows that help data workers conceptually model as a way to reflect on their model implementations and personal biases may be more promising than ones that require them before implementation.

Data workers face challenges obtaining and integrating conceptual and statistical information.

Based on data workers' information search behaviors and self-reports, we found that data workers faced challenges obtaining and integrating both domain and statistical knowledge.

Data workers consulted outside resources such as API documentation, Wikipedia, and the *Towards Data Science* blog throughout the study: one while brainstorming hypotheses [D13]; three while conceptual modeling [D12, D13, D22]; six while specifying statistical models [D3, D6, D12, D13]. Six data workers also mentioned consulting outside resources while implementing their analyses [D1, D3, D11, D14, D15, D21]. By far, statistical help was the most common.

Furthermore, when data workers reflected on their prior data analysis experiences, they detailed how collaborators provided domain and statistical expertise that are instrumental in formalizing hypotheses. Collaborators share data that help domain experts generate hypotheses [D9], critique and revise conceptual models and proxies [D4, D8], answer critical data quality questions [D10], and ensure statistical methods are appropriate [D5, D6, D22].

In the survey participants completed after implementing their analyses, the three most commonly reported challenges were (i) **formatting** the data [D1, D4, D5, D6, D13, D16, D18, D20, D21, D24], (ii) **identifying** which statistical analyses to perform with the data to test their hypotheses [D1, D11, D14, D18, D20, D21], and (iii) **implementing and executing** analyses using their tools [D1, D6, D7, D13, D20, D21]. Although we expected data workers would have difficulty wrangling their data based on prior work Kandel et al. [2012], we were surprised that identifying and executing statistical tests were also prevalent problems given that (a) data workers were relatively experienced and (b) could choose their tools. These results, together with our observations that data workers rely on their prior experiences and tools, suggest that data workers have difficulty adapting to new scenarios where new tools and statistical approaches may be necessary.

4.4.3 Takeaways from the Lab Study

After the first session, 13 out of the 24 data workers described all the tasks as familiar, and 10 described most of the tasks and process as familiar. Data workers commonly remarked that although the process was familiar, the order of the tasks was "opposite" of their usual workflows. In practice, data workers may start with model implementation before articulating conceptual hypotheses, which opposes the direction of data analysis that the ASA recommends Carver et al. [2016]. Nevertheless, our observations reinforce the dual-search, non-linear nature of hypothesis formalization.

Moreover, one data worker, D24, a physics researcher who primarily conducted simulation-based studies expressed that the study and its structure felt foreign, especially because they had no control over data collection. Other data workers in the study also described the importance of designing and conducting data collection as part of their hypothesis formalization process [D4, D6, D9]. Designing data collection methods informs the statistical models data workers plan to use and helps to refine their conceptual hypotheses by requiring data workers to identify proxies and the feasibility of collecting the proxy measures, reinforcing what we saw in the content analysis. The remarks also suggest that disciplines practice variations of the hypothesis formalization process we identify based on discipline-specific data collection norms and constraints. For example, simulating data may sometimes take less time than collecting human subjects data, so data workers working with simulations may dive into modeling and data whereas others may need to plan experiments for a longer period of time.

Approximately half of the data workers had either just finished or were enrolled in undergraduate or graduate programs involving data analysis. As such, half of our sample likely has limited professional experience outside of their studies and/or freelance work on Upwork. Additionally, data work available on Upwork may be more narrowly focused and less representative of end-to-end data analysis or research projects expected of those with greater statistical expertise. Still, several data workers in our study mentioned other employments where they gained professional experience working on larger analysis and research projects. Despite the limitations of recruiting participants from Upwork and word of mouth, our sample represents data workers who have training in a diversity of disciplines (e.g., medicine, psychology, business), are familiar with a range of statistical methods, and have experience using a broad range of statistical tools. As such, the data workers in our study may be representative of analysts who are likely to benefit most from new tools for supporting hypothesis formalization.

Finally, we found that data workers relied on prior experiences and tools to specify and formalize their hypotheses. Tools that scaffold the hypothesis formalization process by suggesting statistical models that operationalize the conceptual hypotheses, conceptual models, or partial specifications data workers create along the way may (i) nudge data workers towards more robust analyses that test their hypotheses, (ii) overcome limitations of data workers' prior experiences, and (iii) even expand data workers' statistical knowledge. Thus, we investigated how current tool designs serve (or under-serve) hypothesis formalization.

4.5 Analysis of Software Tools

To understand how the design of statistical computing tools may support or hinder hypothesis formalization (**RQ3 - Tools**), we analyzed widely used software packages and suites. Throughout, we use the term "package" to refer to a set of programs that must be invoked through code, such as `lme4`, `scipy`, and `statsmodels`. We use the term "suite" to refer to a collection of packages that end-users can access either through code or graphical user interfaces (GUIs), such as SPSS, SAS, and JMP. We use the term "tool" to refer to both. Software packages were a unit of analysis because they are necessary for model implementation regardless of medium (e.g., computational notebook, CoLab, RStudio). As such, our findings apply to tools that provide wrappers around packages included in our sample.

4.5.1 Method

Sample: Our sampling procedure involved two phases: (i) identifying software packages and suites for model implementation (not visual analysis tools like Tableau) mentioned more than once across the content analysis and lab study and (ii) adding recommended packages and suites from online data science communities our lab participants mentioned or used (e.g., *Towards Data Science*). To identify these additional tools, we consulted online data analysis fora Grolemund [2019]; Bobriakov [2017, 2018]; Prabhu [2019]. The final sample included 20 statistical tools: 14 packages (R: 10, Python: 4); three suites that support in-tool programming; and three suites that do not support programming. Table 4.1 contains an overview of our sample and results.

Analysis: Four specific questions guided our analysis:

- **Specialization:** Data workers in the lab study eagerly named specific statistical tools they would use and looked up tool documentation during the tasks. This prompted us to ask, *How specialized are the tools, and how might specialization (or lack thereof) affect how end-users discover and use them to formalize hypotheses?*

- **Statistical Taxonomies:** Data workers in the lab study tried to mold their analyses to prior experiences and their taxonomies of statistical methods. We wondered what role tools play in this: *How do tools organize and group statistical models? How might tool organization and end-users' taxonomies interplay during hypothesis formalization?*

- **Model Expression:** Data workers in the lab study jumped to model implementation throughout the tasks. Only half provided names of statistical methods. We wondered if this was due to how tools enable end-users to express their models: *What notation must end-users use to express models in the tools?*

- **Computational Issues:** Data workers in the lab study described their statistical models using specific function calls. Similarly, although it was uncommon for researchers in the content analysis to specify the software tools they used, when they did, researchers specified the functions, parameters, and settings used. This prompted us to wonder about the importance of computational settings: *What specific kinds of computational control do tools provide end-users and how might that impact hypothesis formalization?*

To answer the four questions for each statistical tool, the first author read and took notes on published articles about tools' designs and implementations, API documentation and reference manuals, and available source code; followed online tutorials; consulted question-and-answer sites (e.g., StackExchange) when necessary; and analyzed sample data with the tools. The first author paid particular attention to tool organization, programming idioms, functions and their parameters, and tool failure cases. Table 4.1 contains citations for resources consulted in the analysis. The iterative analysis process involved discussions among the co-authors about how to evaluate the properties of tools from our perspectives as both tool designers/maintainers and end-users. Here, we focus on end-user (hereafter referred to as analyst) perspectives informed by our lab study and make callouts to details relevant for tool designers.

4.5.2 Findings and Discussion

We discuss our findings in light of our characterization of hypothesis formalization in Figure **??**. We refer to specific steps and transitions in Figure **??** in **boldface**.

Specialization.

Half the tools [T2, T3, T4, T5, T6, T7, T8, T9, T11, T12] in our sample are specialized in the scope of statistical analysis methods they support (e.g., `brms` supports Bayesian generalized linear multilevel modeling). `edgeR` [T3] provides multiple modeling methods but is specialized to the context of biological count data. Such specialized tools are vital to creating a widely adopted statistical computing ecosystem, such as R.

Despite its importance, tool specialization pushes computational concerns higher up the hypothesis formalization process. Specialized tools require analysts to consider computational settings while picking a statistical tool and, possibly, even while mathematically relating their variables. They fuse the last two steps of hypothesis formalization (**Statistical Specification** and **Model Implementation**). Ultimately, specialization requires analysts to have more (i) computational knowledge and (ii) foresight about their model implementations at the cost of focusing on conceptual or data-related concerns early in hypothesis formalization.

One way tool designers minimize the requisite computational knowledge and foresight while providing the benefits of specialized packages — which may be optimal for specific statistical models or data analysis tasks — is to provide micro-ecosystems of packages. For example, R's `tidymodels` Kuhn and Wickham [2020] and `tidyverse` Wickham et al. [2019] create micro-ecosystems that use consistent API syntax and semantics across interoperable packages. They also push analysts towards what the tool designers believe to be best practices, such as the use of the tidy data format Wickham et al. [2014]. Tools that aim to support hypothesis formalization may consider fitting into or creating micro-ecosystems that provide tool support all along the process, focusing analysts on concepts, data, or model implementation at various points.

Statistical taxonomies.

A consequence of tool specialization is the fragmented view of statistical approaches. For example, we observed analysts in the lab study who viewed the analysis as a classification task gravitate towards machine learning-focused libraries, such as `RandomForest` [T9], `Keras` [T11], and

Table 4.1: Overview of the software tools included in our analysis. Half of the tools are specialized for specific modeling use cases. Most tools use mathematical notation (T18–T20 (✓*) even use mathematical notation in their GUIs). Most tools also provide a wide range of computational control although sometimes they require additional packages [T5, T13]. Tool specialization, organization, notation, and computational control focus analysts on model implementation details, sometimes at the expense of focusing on their conceptual hypotheses.

ID	Tool name	Specialized Scope	Mathematical Notation	Computational Control	References
R Packages					
T1	MASS	—	✓	✓	Ripley et al. [2020]
T2	brms	✓	✓	✓	Bürkner et al. [2017]; Bürkner and Buerkner [2016]
T3	edgeR	✓	✓	✓	Chen et al. [2020a,b]
T4	glmmTMB	✓	✓	✓	Brooks et al. [2017]; Magnusson et al. [2020]
T5	glmnet	✓	—	✓(additional)	Friedman et al. [2020]; Hastie and Qian [2014]
T6	lme4	✓	✓	✓	Bates et al. [2014, 2019]
T7	MCMCglmm	✓	✓	✓	Hadfield et al. [2010]; Hadfield [2020]
T8	nlme	✓	✓	✓	Pinheiro et al. [2020]
T9	RandomForest	✓	✓	✓(minimal)	Breiman et al. [2018]
T10	stats (core library)	—	✓	✓	Team and contributors worldwide [2020]
Python Packages					
T11	Keras	✓	—	✓(minimal)	Chollet et al. [2015]
T12	Scikit-learn	✓	—	✓	scikit-learn developers [2020]; Pedregosa et al. [2011]; Buitinck et al. [2013]
T13	Scipy (scipy.stats)	—	—	✓(additional)	Jones et al. [2021a,b,c]
T14	Statsmodels	—	✓	—	Seabold and Perktold [2010]; Perktold et al. [2020]
Suites, with DSLs for programming					
T15	Matlab (Statistics and ML Toolbox)	—	—	✓	The MathWorks

`scikit-learn` [T12]. Because classification can be implemented as logistic regression, any tool that supports logistic regression, such as the core `stats` library in R [T10], provides equally valid, alternative perspectives on the same analysis and hypothesis. However, tools obfuscate these connections and do not aid analysts in considering reasonable statistical models that may be unfamiliar or outside their personal taxonomy. This may explain why analysts adhered to their personal taxonomies during the lab study.

This problem carries over to tools that support numerous statistical methods. Ten tools in our sample intend to provide more comprehensive statistical support [T1, T10, T13, T14, T15, T16, T17, T18, T19, T20]. These tools group statistical approaches using brittle and inconsistent taxonomies based on data types [T17]; analysis classes that are both highly specific (e.g., "Item Response Theory") and vague (e.g., "Multivariate analyses") [T15, T16, T17, T18, T19, T20]; and disciplines or applications (e.g., "Epidemiology and related," "Direct Marketing") [T16, T17, T20]. Although well-intended to simplify statistical method selection, tools' taxonomies are at times misleading. For instance, JMP combines various linear models into a "Fit Model" option that is separate from "Predictive Modeling" and "Specialized Modeling," which are also distinct from the more general "Multivariate Methods." Once analysts select the "Fit Model" option, they can specify the "Personality" of their model as "Generalized Regression," "Generalized Linear Model," or "Partial Least Squares," among many others. This JMP menu structure implies that (i) a Partial Least Squares model is distinct from a regression model when it is in fact a type of regression model and (ii) regression is not useful for prediction, which is not the case.

In these ways, tools add a "Navigate taxonomies" step before the **Statistical Specification** step, requiring analysts to match their conceptual hypotheses with the tools' taxonomies, which may misalign with their personal taxonomies. One reason for this issue may be that tools do not leverage analysts' intermediate artifacts or understanding during hypothesis formalization. By the time analysts transition to **Statistical Specification**, they have refined their conceptual hypotheses, developed causal models, and made observations about data. However, tools' taxonomies require analysts to set these aside and consider another set of decisions imposed by tool-specific groupings of statistical methods. In this way, tool taxonomies may introduce challenges that detract from hypothesis formalization.

Model expression: Syntax and semantics

Fifteen tools in our sample provide analysts with interfaces that use mathematical notation to express statistical models [T1, T2, T3, T4, T6, T7, T8, T9, T10, T14, T16, T17, T18, T19, T20]. R and Python packages use symbolic mathematical syntax, and SPSS and Stata use natural language-like syntax. Expressing a linear model with Sex, Race, and their interaction as predictors of Annual Income involves the formula `AnnualIncome ~ Sex + Race + Sex*Race` in `lme4` and `AnnualIncome BY Sex Race Sex*Race` in SPSS. In a linear execution of steps involved in hypothesis formalization where analysts relate variables mathematically (**Mathematical Equation**) before specifying and implementing models using tools (**Statistical Specification**, **Model Implementation**), the mathematical interfaces match analysts' progression. However, in the lab study, analysts did not specify their models mathematically even when given the opportunity, suggesting that mathematical syntax may not adequately capture analysts' conceptual or statistical considerations.

Syntactic similarity between packages may lower the barrier to trying and adopting new statistical approaches that more directly test hypotheses and therefore benefit hypothesis formalization. At the same time, syntactic similarity may also introduce unmet expectations of semantic similarity. For example, `brms` [T2] uses the same formula syntax as `lme4` [T6], smoothing the transition between linear modeling and Bayesian linear modeling for analysts. However, based on syntactic similarity, analysts may incorrectly assume statistical equivalence in computed model values. For example, in `brms`, the model intercept is the mean of the posterior when all the independent variables are at *their means*, but in `lme4`, the intercept is the mean of the model when all the independent variables are at *zero*.

Conversely, tools introduce syntactic differences between statistical approaches that are for the most part semantically equivalent, which may lead to additional challenges in hypothesis formalization. For instance, an ANOVA with repeated measures and a linear mixed effects model are similar in intent but require two different function calls, one without a formula (e.g., `AnovaRM` in `statsmodels` [T14]) and another with (e.g., `mixedlm` in `statsmodels` [T14]). Even when considering only ANOVA, tools may provide similar syntax but implement different sums of squares procedures for partitioning variance (i.e., Type I, Type II, or Type III).[6] By default, R's `stats` core package [T10] uses Type I, `statsmodels` [T14] uses Type II, and `SPSS` [T16] uses Type III. The three different sum of squares procedures lead to different F-statistics and p-values, which may lead analysts to different conclusions. More importantly, the procedures encode different conceptual hypotheses. If analysts have theoretical knowledge or conceptual hypotheses about the order of independent variables, tools defaulting to Type I (e.g., R's `stats` core library) align the model implementation with the conceptual hypotheses. However, if analysts do not have such conceptual hypotheses, tools' default behavior would execute (without error) and silently respond to a conceptual hypothesis different from the one the analyst seeks to test. In this way, syntactic and

[6] Type I is (a) sensitive to the order in which independent variables are specified because it assigns variance sequentially and (b) allows interaction terms. Type II (a) does not assign variance sequentially and (b) does not allow interaction terms. Type III (a) does not assign variance sequentially and (b) allows interaction terms. For an easy-to-understand blog post, see Korstanje [2019].

semantic mismatches can create a rift between model implementations and conceptual hypotheses. Furthermore, the impact of tools' "invisible" model implementation choices reinforces the interplay between conceptual and model implementation concerns during hypothesis formalization.

Computational issues.

Tools provide end-users with options for optimizers and solvers used to fit statistical models [T1, T2, T4, T6, T7, T8, T10, T11, T13, T16, T18], convergence criteria used for fitting models [T3, T6, T16, T18], and memory and CPU allocation [T2, T5, T12, T15], among more specific customizations. For instance, `lme4` [T6] allows analysts to specify the nonlinear optimizer and its settings (e.g., the number of iterations, convergence criteria, etc.) used to fit models. In `brms` [T2], analysts can also specify the number of CPUs to dedicate to fitting their models. Some computational settings are akin to performance optimizations, affecting computer utilization but not the results. However, not all computational changes are so well-isolated.

For example, the failure of a model's inference algorithm to converge (in **Model Implementation**) may prompt mathematical re-formulation (**Mathematical Equation**), which may cast **Observations about Data** in a new light, prompting **Causal Model** and **Conceptual Hypothesis** revision. In other words, computational failures and decisions may bubble up to conceptual hypothesis revision and refinement, which may then trickle back down to model implementation iteration, and so on. In this way, computational control can be another entry into the dual-search process of hypothesis formalization.

In theory this low-level control could help analysts formalize nuanced conceptual hypotheses in diverse computational environments. However, we found that tools do not currently provide feedback on the ramifications of these computational changes, introducing a gulf of evaluation Norman [1986]. Analysts can easily change parameters to fine-tune their computational settings, but how they should interpret their model implementations and revisions conceptually is unaddressed, suggesting opportunities for future tools to bridge the conceptual and model implementation gap.

4.5.3 Takeaways from the Analysis of Tools

Taken together, our analysis shows that tools can support a wide range of statistical models but expect analysts to have more statistical expertise than may be realistic. They provide limited guidance for analysts (i) to express and translate their conceptual and partially-formalized concerns and (ii) identify reasonable models. Tools also provide little-to-no feedback on the conceptual ramifications of model implementation iterations. These gaps reveal a misalignment between analysts' hypothesis formalization processes and tools' expectations and design. Possible reasons for this mismatch may be that tools do not scaffold or embody the dual-search nature of hypothesis formalization or leverage all the intermediate artifacts analysts may create (e.g., refined conceptual hypotheses, causal models, data observations, partial specifications, etc.) throughout the process.

4.6 Discussion: Design Implications for Statistical Analysis Software

Our findings suggest three opportunities for tools to facilitate the dual-search process and align conceptual hypotheses with statistical model implementations at various stages of hypothesis formalization.

4.6.1 Meta-libraries: Connecting Model Implementations with Mathematical Equations

Specialized tools, although necessary for sophisticated statistical computation, require a steep learning curve. *Meta-libraries* could allow analysts to specify their models in high-level code; execute the models using the appropriate libraries in their knowledge bases; and then output library information, functions invoked, any computational settings used, the mathematical model that is approximated, and the model results. Libraries such as Parsnip Kuhn et al. [2020] have begun to provide a unified higher-level interface that allows analysts to specify a statistical model using more "generically" named functions, parameter names, and symbolic formulae (when necessary). Parsnip then compiles and invokes various library-specific functions for the same statistical model.

Probabilistic programming languages (PPLs), such as Pyro **?**, Stan Carpenter et al. [2017], BUGS Lunn et al. [2000], PyMC **?**, already enable the development of meta-libraries. PPLs support modular specification of data, probabilistic models, and probabilistic hypotheses. Existing libraries, including `brms`, provide higher-level APIs whose syntax uses symbolic formulae, for instance, and compile to programs in a PPL (i.e., Stan in the case of `brms`).

As already seen in Parsnip and tools using PPLs, meta-libraries could bring three benefits. First, they would provide simpler, less fragmented interfaces to analysts while continuing to take advantage of tool specialization. Second, meta-libraries that output complete mathematical representations would more tightly couple mathematical representations with implementations, providing an on-ramp for analysts to expand their statistical knowledge. Third, meta-libraries that show the mathematical representations alongside underlying libraries' function calls could show syntactical variation in underlying libraries, indirectly teaching analysts how they might express their statistical models in other tools, familiarizing analysts with new tools and models, and even mend fragmented views of identical models (e.g., ANOVA and regression).

Future meta-libraries could consider providing a higher-level, declarative interface that does not require analysts to write symbolic formulae. Designing such declarative meta-libraries would require formative elicitation studies (similar to natural programming studies such as Verou et al. [2018]) on declarative primitives that are memorable, distinguishable, and reliably understood. An additional challenge would lie in maintaining support for various libraries executed under the hood, especially as libraries change their APIs, which would strengthen the case for meta-libraries. Although meta-libraries would not solve the problems involved in understanding how computational settings affect model execution or conceptual hypotheses, they could nevertheless provide scaffolding for analysts to more closely examine specific libraries, especially if multiple libraries execute the same model but do not all encounter the same computational bottlenecks.

4.6.2 High-level Libraries: Expressing Conceptual Hypotheses to Bootstrap Model Implementations

The absence of tools for directly expressing conceptual hypotheses may be an explanation for why data workers in the lab study dove into model implementation details. High-level libraries could allow analysts to specify data collection design (e.g., independent variables, dependent variables, controlled effects, possible random effects); variable data types; expected or known covariance relationships based on domain expertise; and hypothesized findings in a library-specific grammar. High-level libraries could compile these conceptual and data declarations into weighted constraints that represent the applicability of various statistical approaches, in a fashion similar to Tea Jun et al. [2019], a domain-specific language for automatically selecting appropriate statistical analyses for common hypothesis tests. Libraries could then execute the appropriate statistical approaches, possibly by using a meta-library as described above.

In addition to questions of how to represent a robust taxonomy of statistical approaches computationally, another key challenge for developing high-level libraries is identifying a set of minimal yet complete primitives that are useful and usable for analysts to express information that is usually expressed at different levels of abstraction: conceptual hypotheses, study designs, and possibly even partial statistical model specifications. For instance, even if a conceptual hypothesis is expressible in a library, it may be impossible to answer with a study design or partial statistical model that is expressed in the same program. An approach may be to draw upon and integrate aspects from existing high-level libraries and systems that aim to address separate steps of the hypothesis formalization process, such as Touchstone2 Eiselmayer et al. [2019] for study design and Tea and Statsplorer Wacharamanotham et al. [2015] for statistical analysis.

4.6.3 Bidirectional Conceptual Modeling: Co-authoring Conceptual Models and Model Implementations

Conceptual, or causal, modeling was difficult for the analysts in the lab study. Some even resisted conceptual modeling for fear of biasing their analyses. Yet, implicit conceptual models were evident in the hypotheses analysts chose to implement and the sub-hypotheses researchers articulated in the content analysis.

Mixed-initiative systems that make explicit the connection between conceptual models and statistical model implementations could facilitate hypothesis formalization from either search process and allow analysts to reflect on their analyses without fear of bias. For example, a mixed-initiative programming environment could allow analysts to write an analysis script, detect data variables in the analysis scripts, identify how groups of variables co-occur in statistical models, and then visualize conceptual models as graphs where the nodes represent variables and the edges represent relationships. The automatically generated conceptual models would serve as templates that analysts could then manipulate and update to better reflect their internal conceptual models by specifying the kind of relationship between variables (e.g., correlation, linear model, etc.) and assigning any statistical model roles (e.g., independent variable, dependent variable). As analysts update the visual conceptual models, they could evaluate script changes the system proposes. In this way, analysts could externally represent their causal models while authoring analysis scripts and vice versa.

Although bidirectional programming environments already exist for vector graphics creation Hempel et al. [2019], they have yet to be realized in mainstream data analysis tools. To realize bidirectional, automatic conceptual modeling, researchers would need to address important questions about (i) the visual grammar, which would likely borrow heavily from the causal modeling literature; (ii) program analysis techniques for identifying variables and defining co-occurrences (e.g., line-based vs. function-based) in a way that generalizes to multiple statistical libraries; and (iii) adoption, as analysts who may benefit most from such tools (likely domain non-experts) may be the most resistant to tools that limit the number of "insights" they take away from an analysis.

4.7 Discussion: Data analysis as problem solving

Hypothesis formalization is a dual-search process of translating conceptual hypotheses into statistical model implementations. Due to constraints imposed by domain expertise, data, and tool familiarity, the same conceptual hypothesis may be formalized into different model implementations. A single model implementation may be useful for making multiple statistical inferences. The same model implementation may also formalize two possibly opposing hypotheses. To navigate these constraints, analysts use problem-solving strategies characteristic of the larger scientific discovery process Klahr and Dunbar [1988]; Schunn and Klahr [1995]. As such, hypothesis formalization exemplifies how data science is a design practice.

At a conceptual level, hypothesis formalization involves *hypothesis refinement*, which, to use Schunn and Klahr's language Schunn and Klahr [1995], is a *scoping* process. In the formative content analysis, we found that researchers *decomposed* their research goals and conceptual hypotheses into specific, testable sub-hypotheses and *concretized* constructs using proxies, born of theory or available data. Also, we found that analysts in the lab study also quickly converged on the need to specify established proxies or develop them based on the data schema presented. In hypothesis formalization, scoping incorporates domain- and data-specific observations to qualify the conceptual scope of researchers' hypotheses. In other words, hypothesis refinement is an instance of *means-end analysis* Newell et al. [1972], a problem-solving strategy that aims to recursively change the current state of a problem into sub-goals (i.e., increasingly specific objectives) in order to apply a technique (i.e., a particular statistical model) to solve the problem (i.e., test a hypothesis).

At the other computational endpoint of hypothesis formalization, *model implementation* also involves iteration. Through our analysis of software tools, we found that analysts must not only select tools among an array of specialized and general choices but also navigate tool-specific taxonomies of statistical approaches. These tool taxonomies may both differ from and inform analysts' personal categorizations, potentially explaining why analysts in our lab study relied on their personal taxonomies and tools. Based on their prior experience, analysts engage in *analogical reasoning* Holland et al. [1989], finding parallels between the present analysis problem's structure and previously encountered ones or ones that fit a tool's design easily.

Upon selecting a statistical function, analysts may tune computational settings, choose different statistical functions or approaches, which they may tune, and so on. In this way, the model implementation loop in hypothesis formalization captures the "debugging cycles" analysts encounter, such as the census researcher in the introduction. The tool ecosystem as a whole supports diverse model implementations, even for the same mathematical equation. However, the tool interfaces provide low-level abstractions, such as interfaces using mathematical formulae that, based on our observations in the lab study, do not support the kind of higher-level conceptual reasoning required of hypothesis formalization.

4.8 Future Work

The steps, considerations, and strategies we have identified are domain-general. Domain-specific expertise likely influences how quickly analysts switch between steps and strategies during the dual-search process. Domain experts, including researchers in our content analysis, may know which statistical model implementations and computational settings to use a priori and design their studies or specify their conceptual hypotheses in light of these expectations — incorporating means-end analysis and analogical reasoning strategies — more quickly. It may be these insights that analysts in our lab study sought when they looked online for conceptual and statistical help.

Future work could observe how domain experts perform hypothesis formalization and characterize when and how analysts draw upon their own or collaborators' expertise to circumvent iterations or justify early scoping decisions. These insights may also shed light on how pre-registration expectations and practices could be made more effective. Given the level of detail required of some pre-registration policies, researchers likely engage in a version of the hypothesis formalization process we have identified prior to registering their studies. Knowing how pre-registration fits into the hypothesis formalization process could improve the design and adoption of pre-registration practices.

Future work could also explore how hypothesis formalization may differ in machine learning settings. In this paper, our focus was on how analysts answer domain questions and test hypotheses using statistical methods and their domain knowledge. Our findings may not generalize to settings or methods where domain knowledge is less important, such as deep learning and other machine learning-based approaches.

Finally, our findings suggest opportunities for future tools to bridge steps involved in hypothesis formalization and guide analysts towards reasonable model implementations. Our analysis of tools suggest possibilities for tools to connect model implementations to their mathematical representations through meta-libraries, provide higher-level abstractions for more directly expressing conceptual hypotheses, and support automated conceptual modeling. Future system development and user testing are necessary to validate these implications and more readily support analysts translate their conceptual hypotheses into statistical model implementations.

4.9 Summary of Contributions

The empirical studies that led us to articulate the theory of hypothesis formalization illustrates the key challenge to authoring data analyses: Analysts must translate their implicit domain knowledge into statistical specifications that they can implement and execute in code. As we saw in the lab study, analysts often resort to changing their hypotheses or research questions to what they can implement or get stuck on how to represent their conceptual knowledge in statistical models, highlighting the dual-search nature of hypothesis formalization. Furthermore, the summary of hypothesis formalization (i.e., Figure **??**) serves as a device for (i) interpretation–to explain where and how analysts struggle in authoring statistical analyses–and (ii) inspiration–to inspire new approaches and systems to authoring data analyses.

Our theory of hypothesis formalization highlights the discrepancy between analysts' goals and the statistical software tools available to them. While analysts want to understand their data to better understand their domains or make decisions, the current ecosystem prioritizes mathematical expressivity and computational control, features that are likely desirable for statistical experts but not novices.

As a result, designing new data analysis tools to gather conceptual knowledge and translate them into statistical analyses is a promising approach for statistical non-experts. In this way, hypothesis formalization retrospectively validates our design in Tea, where its constraint-based runtime system provided automated reasoning for Null Hypothesis Significance Tests. In order to support more complex research questions, additional methods of explicitly grappling with more conceptual knowledge and reasoning about different classes of statistical analyses is necessary. We tackle this challenge for generalized linear models with and without mixed effects in Tisane.

This work was in collaboration with Nicole de Moura, Melissa Birchfield, Jeffrey Heer, and René Just. It was originally published in ACM Transactions of Computer-Human Interaction (TOCHI) 2022 and presented at ACM CHI 2022.

Chapter 5

Tisane: Authoring statistical models via formal reasoning from conceptual and data relationships

If you are copying and pasting material from one of your papers, then remember to:

- Consider rephrasing conference-paper-style language:
  - Find every place you mention some variation of “in this paper” and say “in this chapter” instead.
  - Remove or rephrase the parts where you talk about “our main contributions”.
  - Rephrase the language describing code and data releases.

Authoring statistical models requires analysts to jointly reason about their conceptual domain knowledge, statistical methods, and analysis implementations in code, as our theory of hypothesis formalization describes. For instance, scientists carefully consider which covariates to include in statistical models based on their prior knowledge of confounding. However, analysts’ conceptual knowledge is often kept implicit. Analysts gravitate towards statistical specifications they are familiar with, even if the analyses are sub-optimal or do not assess their hypotheses, as we saw in the previous chapter. Finally, ease of implementation further constrains the statistical models that analysts try and use. These issues are especially salient for domain experts who lack deep statistical or programming expertise (e.g., many researchers).

Existing statistical software exacerbate these issues because they do not allow analysts to externalize their implicit conceptual knowledge, receive guidance on analysis approaches, and help authoring low-level statistical modeling code Section 4.5. Our work on Tisane hypothesizes that in order to address these issues, software tools should capture analysts' implicit conceptual models and use them to derive statistical models.

Conceptual models [1] are often-informal representations of variable relationships (e.g., list of variable relationships, process diagrams, graphs), describing the underlying data generating process. Conceptual models are difficult to reason about during statistical analysis. Their implications on statistical modeling are not obvious, especially to statistical non-experts. For example, the impact of conceptual assumptions may only become apparent after fitting multiple statistical models, if at all. Without explicitly grappling with conceptual models prior to authoring statistical models, analysts run the risk of introducing inconsistencies between their domain knowledge and statistical models, which can lead to unintentionally answering a different research question and asserting a conceptual model based on preferred results (i.e., HARKing).

To facilitate more accurate hypothesis formalization and analysis, we asked, **How might we derive (initial) statistical models from conceptual models?**. Inferring a statistical model raises two technical challenges: (1) How do we elicit the information necessary for inferring a statistical model? and (2) How do we infer a statistical model, given this information? We explore and address these issues by iteratively designing, developing, and evaluating **Tisane, a system for implementing generalized linear models (GLMs) and generalized linear mixed-effects models (GLMMs) from explicit statements of implicit conceptual assumptions**.

The first implementation of Tisane (Section 5.3) was as an open-source Python package available on `pip` **?**. Case studies and real-world use of Tisane demonstrated not only the viability but also the desirability of tool support for authoring statistical models from conceptual models. Therefore, we explored how to further improve Tisane's programming and interaction models to better suit novice analysts (Section 5.6) and released a second version in R as the rTisane library **?**. The R implementation allowed us to more direclty compare rTisane to a scaffolded workflow using widely used linear modeling libraries, including the `lme4`, in R.

Tisane provides a **study design specification language** for expressing relationships between variables. Tisane compiles the explicitly stated relationships into an internal **graph representation** and then traverses the graph to infer candidate GLMs/GLMMs based on recommendations from the graphical causal reasoning community. Analysts can then query Tisane for a statistical model that explains a specific dependent variable from a independent variable of interest. Tisane helps analysts disambiguate their input conceptual models and an output statistical model script for fitting a valid GLM/GLMM. In this way, Tisane focuses analysts on reflecting on and externalizing their implicit conceptual assumptions and checks that analysts do not overlook relevant variables, such

[1] Richard McElreath calls these implict assumptions *process models* McElreath [2020]. We use the term *conceptual models* in order to contrast from statistical models.

as potential confounders or data clustering, that could compromise generalizability of statistical results (*DI1 - Raise level of abstracion*, *DI2 - Connect conceptual and statistical models*).

To do (??)

5.0.1 Statistical scope

5.1 Why is Tisane necessary, isn't Tea enough?

Many research questions analysts want to answer require more complex analyses. GLMs and GLMMs are meaningful targets because they are commonly used (e.g., in psychology Lo and Andrews [2015]; Cohen et al. [2013], social science Kreft et al. [1998], and medicine Bolker et al. [2009]; Barr et al. [2013]) yet are easy to misspecify for statistical experts and non-experts alike Barr et al. [2013]; Cohen et al. [2013]. We designed Tisane to support researchers who are domain experts capable of supplying conceptual and data collection information but lack the statistical expertise or confidence to author GLM/GLMMs accurately. Both GLMs and GLMMs consist of (i) a *model effects structure*, which can include main and interaction effects and (ii) *family* and *link* functions. The family function describes how the residuals of a model are distributed. The link function transforms the predicted values of the dependent variable. This allows modeling of linear and non-linear relationships between the dependent variable and the predictors. In contrast to transformations applied directly to the dependent variable, a link function does not affect the error distributions around the predicted values. The key difference between GLMs and GLMMs is that GLMMs contain random effects in their model effects structure. Random effects describe how individuals (e.g., a study participant) vary and are necessary in the presence of hierarchies, repeated measures, and non-nesting composition (5.3.1)[2].

Both GLMs and GLMMs assume that (i) the variables involved are linearly related, (ii) there are no extreme outliers, and (iii) the family and link functions are correctly specified. In addition, GLMs also assume that (iv) the observations are independent. Tisane's interactive compilation process guides users through specifying model effects structures, family and link functions to satisfy assumption (iii), and random effects only when necessary to pick between GLMs and GLMMs and satisfy assumption (iv).

In the scope of this thesis, GLM and GLMMs are an appropriate scope because they encompass a large scope of statistical models such that our research contributions are widely applicable and substantial. In addition, given that NHSTs (in Tea) are mathematically related to GLMs and GLMMs, Tisane's focus helps us to push the boundaries of the applicability of higher-level abstrac-

[2]Traditionally, the term "mixed effects" refers to the simultaneous presence of "fixed" and "random" effects in a single model. We try to avoid these terms as there are many contradictory usages and definitions Gelman [2005]. When we do use these terms, we use the definitions from Kreft and De Leeuw Kreft et al. [1998].

tions for statistical analysis established/explored in Tea and lay the groundwork to connect Tea and Tisane's programming and interaction models (better design statistical computing tools accurately reflect statistical taxonomies).

We output models, not causal measures/estimates because people might want to revise, iterate on. They also expect a model, so Tisane is just a step towards moving people towards more sophisticaed estimates?

Recent work in the database community helps researchers answer causal questions about multilevel, or hierarchical, data Salimi et al. [2020]; Kayali et al. [2020]. CaRL Salimi et al. [2020] provides a domain-specific language to express causal relationships between variables and a GUI to show researchers results. Tea and Tisane leverage a similar insight that researchers have domain knowledge that a system can use to infer statistical methods. Whereas CaRL is focused on answering specific queries about average causal effect, the systems in this dissertation are designed to address a range of non-causal questions as well.

5.2 Background and Related work

5.2.1 Causal Analysis

One of our aims in Tisane was to connect conceptual models to statistical models (*DI2 - Connect conceptual and statistical models*). Prior work in the causal reasoning literature shows how linear models can be derived from causal graphs to make statistical inferences and test the motivating causal graph Spirtes et al. [1996]; Spirtes [1994]. Recently, VanderWeele proposed the "modified disjunctive cause criterion" VanderWeele [2019] as a new heuristic for researchers without a clearly accepted formal causal model to identify confounders to include in a linear model, for example. The criterion identifies confounders in a graph based on expressed causal relationships. Tisane applies the modified disjunctive cause criterion when suggesting variables to include in a GLM or GLMM. Tisane does not automatically include variables to the statistical models because substantive domain knowledge is necessary to resolve issues of temporal dependence between variables, among other considerations VanderWeele [2019]. To guide analysts through the suggestions, Tisane provides analysts with explanations to aid their decision making during disambiguation. Finally, GLMs are not formal causal analyses. Tisane does not calculate average causal effect or other causal estimands. Rather, Tisane only utilizes insights about the connection between causal DAGs and linear models to guide analysts towards including potentially relevant confounders in their GLMs grounded in domain knowledge.

There are multiple frameworks for reasoning about causality Rubin [2004]; Pearl [1995a]. One widespread approach is to use directed acyclic graphs (DAGs) to encode conditional dependencies between variables Pearl [1995b]; Greenland et al. [1999]; Spirtes [1994]; Spirtes et al. [1996]. If analysts can specify a formal causal graph, Pearl's "backdoor path criterion" Pearl [1995a]; Pearl et al. [2000] explains the set of variables that control for confounding. However, in practice, specifying proper causal DAGs is challenging and error-prone for domain experts who are not also experts in causal analysis Suzuki et al. [2020] due to uncertainty of empirical findings Suzuki and VanderWeele [2018] and lack of guidance on which variables and relationships to include Velentgas et al. [2013]. Accordingly, Tisane does not expect analysts to specify a formal causal graph. Instead, analysts can express causal relationships as well as "looser" association (not causal) relationships between variables in the study design specification language.

5.3 First Release

This work was originally published at ACM CHI, where it received a *Best Paper Honorable Mention awward.*

5.3.1 Study design specification language and graph representation

Tisane provides a *study design specification language* (*SDSL*) for expressing relationships between variables. There are two key challenges in designing a specification from which to infer statistical models: (1) determining the set of relationships that are essential for statistical modeling and (2) determining the level of granularity to express relationships.

In Tisane's SDSL, analysts can express conceptual and data measurement relationships between variables. Both are necessary to specify the domain knowledge and study designs from which Tisane infers statistical models.

Variables

There are three types of data variables in Tisane's SDSL: (i) units, (ii) measures, and (iii) study environment settings. The **Unit** type represents entities that are observed and/or receive experimental treatments. In the experimental design literature, these entities are referred to as "observational units" and "experimental units," respectively. Entities can be both observational and experimental units simultaneously, so the SDSL does not provide more granular unit sub-types. The **Measure** type represents attributes of units and must be constructed through their units, e.g., `age = adult.numeric('age')`. Measures are proxies (e.g., minutes ran on a treadmill) of underlying constructs (e.g., endurance). Measures can have one of the following data types: numeric, nominal, or ordinal. Numeric measures have values that lie on an interval or ratio scale (e.g., age, minutes ran on a treadmill). Nominal measures are categorical variables without an ordering (e.g., race).

Ordinal measures are ordered categorical variables (e.g., grade level in school). We included these data types because they are commonly taught and used in data analysis. The **SetUp** type represents study environment settings that are neither units nor measures. For example, time is often an environmental variable that differentiates repeated measures but is neither a unit nor a measure of a specific unit.

Relationships between Variables

In Tisane's SDSL, variables have relationships that fall into two broad categories: (1) *conceptual relationships* that describe how variables relate theoretically and (2) *data measurement relationships* that describe how the data was, or will be, collected. Below, we define each of the relationships in Tisane' SDSL and describe how Tisane internally represents these relationships as a graph (as illustrated in 5.3.1). **??** shows the graph representation constructed from the usage scenario.

Tisane's graph IR is a directed multigraph. Nodes represent variables, and directed edges represent relationships between variables. Tisane internally uses a graph intermediate representation (IR) because graphs are widely used for both conceptual modeling and statistical analysis, two sets of considerations that Tisane unifies.

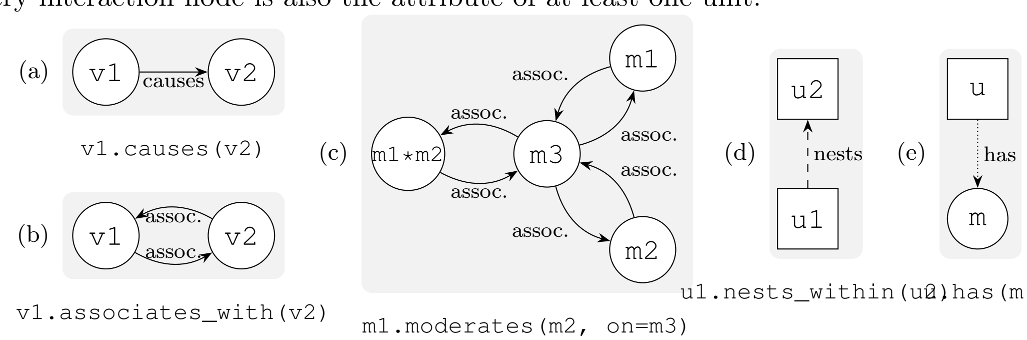
Tisane's graph IR differs from two types of graphs used in data analysis: causal DAGs and path analysis diagrams. Unlike causal DAGs, Tisane's graph IR allows for non-causal relationships, moderating relationships (i.e., interaction effects), and data measurement relationships that are necessary for inferring random effects. Unlike path analysis diagrams that allow edges to point to other edges to represent interaction effects, Tisane represents interactions as separate nodes and only allows nodes as endpoints for edges. These design decisions simplify our statistical model inference algorithms and their implementation.

Conceptual relationships. Tisane's SDSL supports three conceptual relationships: causes, associates with, and moderates. Analysts can express that a variable **causes** or is **associated with** (but not directly causally related to) another variable. Variables associated with the dependent variable, for example, may help explain the dependent variable even if the causal mechanism is unknown. If analysts are aware of or suspect a causal relationship, they should use `causes`.

We chose to support both causal and associative relationships because formal causal DAGs are difficult for domain experts to specify Suzuki et al. [2020]; Suzuki and VanderWeele [2018]; Velentgas et al. [2013], prior work has observed that researchers already use informal graphs that contain associative relationships when reasoning about their hypotheses and analyses Jun et al. [2022], and GLMs/GLMMs can represent non-causal relationships. Finally, analysts can also express interactions where one (or more) variable (the *moderating variables*) **moderates** the effect of a *moderated variable* on another variable (the *target variable*).

Mediation relationships (where one variable influences another through a middle variable) are another common conceptual relationship. Tisane does not provide a separate language construct for mediation because mediations are expressible using two or more causal relationships. Furthermore, mediation analyses require specific analyses, such as structural equation modeling Hoyle [1995], that are out of Tisane's scope.

In the graph IR, a `causes` relationship introduces a causal edge from one node, the cause, to another node, the effect (5.3.1(a)). Because a variable cannot be both the cause and effect of the same variable, any pair of nodes can only have one causal edge between them. Furthermore, from a formal causal analysis perspective, associations may indicate the presence of a hidden, unobserved variable that mediates the causal effect of a variable on another or that influences two or more variables simultaneously. Thus, rather than inferring or requiring analysts to specify hidden variables, which may be unknown and/or unmeasurable, the `associates_with` relationship introduces two directed edges in opposing directions, representing the bidirectionality of association (5.3.1(b)). A `moderates` relationship creates a new node that is eventually transformed into an interaction term in the model, introduces associative edges between the new interaction node and the target (variable) node, creates associative edges between the moderated variable's node and the target node, and adds associative edges between the moderating variables' nodes and the target node if there is not a causal or associative edge already (5.3.1(c)). Furthermore, each interaction node inherits the attribution edges from the nodes of the moderating variables that comprise it. This means that every interaction node is also the attribute of at least one unit.[3]



Data measurement relationships. Study designs may have clusters of observations that need to be modeled explicitly for external validity. For example, in a within-subjects experiment, participants provide multiple observations for different conditions. An individual's observations may cluster together due to a hidden latent variable. Such clustering may be imperceptible during ex-

[3]In statistical terms, this means that within-level interactions have one unit while cross-level interactions may have two or more units.

ploratory data visualization of a sample but can threaten external validity. GLMMs can mitigate three common sources of clustering that arise during data collection Gelman and Hill [2006]; Kreft et al. [1998]; Cohen [1988]:

- **Hierarchies** arise when one observational/experimental unit (e.g., adult) nests within another observational/experimental unit (e.g., group). This means that each instance of the nested unit belongs to one and only one nesting unit (many-to-one).
- **Repeated measures** introduce clustering of observations from the same unit instance (e.g., participant).
- **Non-nesting composition** arises when overlapping attributes (e.g., stimuli, condition) describe the same observational/experimental unit (e.g., participant) Gelman and Hill [2006].

The above sources of clustering pose three problems for analysts. First, analysts must have significant statistical expertise to identify when data observations cluster. Second, they must know how to mitigate these clusters in their models. Third, with this knowledge, analysts must figure out how to express these types of clustering in their analytical tools. Even if analysts are not able to identify clustered observations, they are knowledgeable about how data were collected.

Thus, Tisane addresses the three problems by (i) eliciting data measurement relationships from analysts to infer clusters and (ii) formulating the maximal random effects structure, optimizing for external validity (5.3.2). Below, we describe language features for expressing data measurement relationships.

Nesting relationships: Hierarchies **Hierarchies** arise when a unit (e.g., an `adult`) is nested within another unit (e.g., an exercise `group`). Researchers may collect data with hierarchies to study individual and group dynamics together or as a side effect of recruitment strategies. To express such designs, Tisane provides the `nests_within` construct. Conceptually, nesting is strictly between observational/experimental units, so Tisane type checks that the variables that nest are both Units. In the graph IR, a nesting relationship is encoded as an edge between two unit nodes (5.3.1(d)). There is one edge from the nested unit (e.g., `adult`) to the nesting unit (e.g., `group`) [4].

Frequency of measures: Repeated measures, Non-nesting composition When a measure is declared through a unit, Tisane adds an attribution edge ("has') from a unit node to a measure node (5.3.1(e)). A unit's measure can be taken one or more times in a study. The frequency of measurement is useful for detecting repeated measures and non-nesting composition. In **repeated**

[4]The GitHub repo contains a gallery of examples that include nesting relationships.

measures study designs, each unit provides multiple values of a measure, which are distinguished by another variable, usually time. **Non-nesting** Gelman and Hill [2006] composition arises when measures describing the same unit overlap. For example, HCI researchers studying input devices might design them to utilize different senses (e.g., touch, sight, sound). Participants in the study may be exposed to multiple different devices, which act as experimental conditions of senses. The conditions are intrinsically tied to the devices, and participants can be described as having both conditions and devices, which overlap with one another. Such study designs introduce dependencies between observations Clark [1973] and hence violate the assumption of independence that GLMs make.

When analysts declare Measures, they specify the frequency of the observation through the `number_of_instances` parameter. This parameter accepts an integer, variable, a Tisane `Exactly` operator, or a Tisane `AtMost` operator. By default, the parameter is set to one. The `Exactly` operator represents the exact number of times a unit has a measure. The `AtMost` operator represents the maximum number of times a unit has a measure. Both operators are useful for specifying that a measure's frequency depends on another variable, which is expressible through the `per` function. For example, participants may use two `devices` *per* `condition` assigned: `device = subject.nominal('Input device', number_of_instances=ts.Exactly(2).per(condition))`. The `per` function uses the Tisane variable's cardinality by default but can instead use a data variable's `number_of_instances` by specifying `use_cardinality=False` as a parameter to `per`. Moreover, specifying a measure's `number_of_instances` to be an integer is syntactic sugar for using the `Exactly` operator. Specifying a variable is syntactic sugar for expressing `ts.Exactly(1).per(variable)`.

To determine the presence of repeated measures or non-nesting composition, Tisane computes the `number_of_instances` of measures and their relationship to other measures. Measures that are declared with `number_of_instances` equal to one are considered to vary between-unit. Measures that are declared with `number_of_instances` greater than one or a variable with cardinality greater than one are considered to vary within-unit as repeated measures. If there are instances of a measure per another measure sharing the same unit, the measures are non-nesting.

5.3.2 Statistical model inference: Interactively querying the graph IR

After specifying variable relationships, analysts can query Tisane for a statistical model. Queries are constructed by specifying a study design with a dependent variable (the value to be predicted) and a set of independent variables (predictors). Tisane processes the query and generates a statistical model in four phases: (1) preliminary conceptual checks that validate the study design, (2) inference of possible effects structures and family and link functions, (3) input elicitation to disambiguate possible models, and (4) generation of a final executable script, and a record of decisions during disambiguation. Given that the interactive process begins with an input program using Tisane and outputs a script for fitting a GLM or GLMM, we call this process *interactive compilation*.

Preliminary checks

At the beginning of processing a query, Tisane checks that every input study design is well-formed. This involves two conceptual correctness checks. First, every independent variable (IV) in the study design must either cause or be associated with the dependent variable (DV) directly or transitively. Second, the DV must not cause any of the IVs, since it would be conceptually invalid to explain a cause from any of its effects. If any of the above checks fail, Tisane issues a warning and halts execution. By using these two checks, the Tisane compiler avoids technically correct statistical models that have little to no conceptual grounding (*DG1 - Conceptual knowledge*). If the checks pass, Tisane proceeds to the next phase.

Candidate statistical model generation

A GLM/GLMM is comprised of a model effects structure, family function, and link function. The model effects structure may consist of main, interaction, and random effects. Tisane utilizes variables' conceptual relationships to infer candidate main and interaction effects and data measurement relationships to infer random effects. Tisane infers family and link functions based on the data type of the DV in the query. The candidate statistical models that Tisane generates, based on the graph and query, seed an interactive disambiguation process.

The purpose of identifying candidate main effects beyond the ones analysts may have specified is to provoke consideration of erroneously omitted variables that are conceptually relevant and pre-empt potential confounding and multicollinearity issues that may arise.

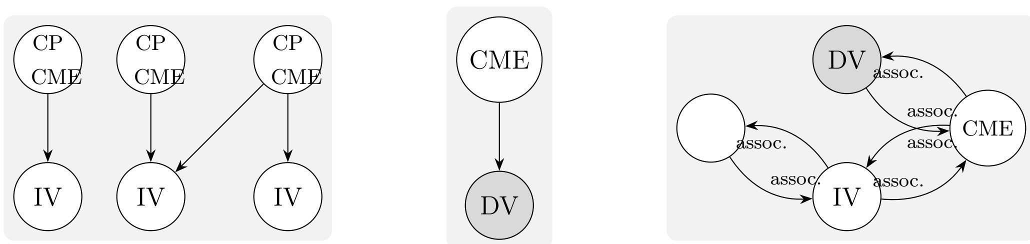
Deriving Candidate Main Effects In a query to infer a statistical model, analysts specify a single dependent variable and a set of one or more IVs. After passing the checks described in 5.3.2, the query's independent variables are considered candidates. In addition, Tisane derives three additional sets of candidate main effects intended to control for confounding variables in the output statistical model[5]. The first two sets below are from the "modified disjunctive cause criterion" VanderWeele [2019]:

- **Causal parents.** For each IV in the query, Tisane finds its causal parents (see 5.3.2(a)).

[5]Tisane currently treats each input IV as a separate "exposure" variable for which to identify confounders. Tisane then combines all confounders into one statistical model.

- **Possible causal omissions.** Tisane looks to see if any other variables not included as IVs cause the DV (see in 5.3.2(b)). They are relevant to the DV but may have been erroneously omitted.

- **Possible confounding associations.** For each IV, Tisane looks for variables that are associated with both the IV and the DV (see in 5.3.2(c)). Because associations between variables can have multiple underlying causal structures, Tisane recommends variables with associative relationships with caution. Tisane issues a warning describing when not to include such a variable in the GUI.



(a) *causal parents* (b) *possible causal omissions* (c) *possible confounding associations*

Using the above rules, Tisane suggests a set of variables that are likely confounders of the variables of interest expressed in the query. There may be additional confounders due to unmeasured or unexpressed variables that are either not known or excluded from the graph. Tisane never automatically includes the candidate main effects in the output statistical model. Analysts must always specify a variable as an IV in the query or accept a suggestion (*DG3 - Guidance and control*).

If a graph only contains associates edges then the candidate main effects Tisane suggests are those that are directly associated with both the DV and an IV. If a graph has only causal edges, Tisane would suggest variables that directly cause the DV but were omitted from the query and the causal parents of IVs in case the parents exert causal influence on the DV through the IV or another variable that is not specified.

The total set of main effects, including variables the analyst has specified as IVs in their query and candidate main effects, are used to derive candidate interaction effects and random effects, which we discuss next.

Deriving Candidate Interaction Effects An interaction between variables means that the effect of one variable (the *moderated* variable) on a *target* variable is moderated by another (non-empty) set of variables (the *moderating* variables). Tisane's SDSL already provides a primitive, `moderates`, to express interactions. As such, Tisane's goal in suggesting candidate interaction effects is to help analysts avoid omissions of conceptual relationships that are pertinent to an analyst's research questions or hypotheses (*DG1 - Conceptual knowledge*). Candidate interaction effects are the interaction nodes whose (i) moderated and moderating variables include two or more candidate main effects and (ii) target variable is the query's DV.

Deriving Candidate Random Effects Random effects occur when there are clusters in the data, which occur when we have repeated measures, nested hierarchies, or non-nesting composition (as defined in subsection 5.3.1). Tisane implements Barr et al.'s recommendations for specifying the maximal random effects structure of linear mixed effects models for increasing the generalizability of statistical results Barr et al. [2013]; Barr [2013].

To derive random effects, Tisane focuses on the data measurement edges in the graph IR. Using the graph IR, Tisane identifies unit nodes, looks for any nesting edges among them, and determines within- or between-subjects measures based on the frequency of observations for units. From these, Tisane generates random intercepts of units for the unit's measures that are between-subjects as well as the unit's measures that are within-subjects where each instance of the unit has only one observation per value of another variable. Tisane generates random slopes of a unit and its measure for all measures that are within-subjects where each instance of the unit has multiple observations per value of another variable. For interaction effects, random slopes are included for the largest subset of within-subjects variables (see Barr [2013]). Tisane handles correlation of random slopes and intercepts during disambiguation (subsection 5.3.2). Maximal random effects may lead to model convergence issues that analysts address by later removing or adding independent variables and random effects. Nevertheless, starting with a maximal, valid model is important for ensuring that future revisions are also valid (*DG2 - Validity*).

Deriving Candidate Family and Link Functions The DV's data type determines the set of candidate family and link functions. For example, numeric variables cannot have binomial or multinomial distributions. Similarly, nominal variables are not allowed to have Gaussian distributions. Furthermore, each family has a set of possible link functions. For example, a Gaussian family distribution may have an Identity, Log, or Square Root link function. The statistics literature documents possible combinations of family and link functions for specific data types Nelder and Wedderburn [1972].

Tisane includes common family distributions as candidate families and their applicable link functions. In its current implementation, Tisane relies on `statsmodels` Seabold and Perktold [2010] for GLMs and `pymer4` Jolly [2018] for GLMMs. As such, Tisane is limited to the family and link function pairings implemented in these libraries. As `statsmodels`' and `pymer4`'s support for GLMs grows in the future, Tisane can be extended.

Eliciting Analyst Input for Disambiguation

The disambiguation process provides an opportunity for analysts to explore the space of generated models based on their original query. Given our design considerations to prioritize conceptual knowledge (*DG1 - Conceptual knowledge*) and give analysts guidance (*DG3 - Guidance and control*), we designed a GUI to scaffold analysts' reasoning and elicit their input. For versatility, we implemented Tisane's GUI using Plotly Dash Community [[n. d.]]. Analysts can either execute their Tisane programs and use the GUI inside a Jupyter notebook (no additional widgets needed) or run their Tisane programs in an IDE or terminal, in which case Tisane will open the GUI in a web browser.

Candidate statistical models are organized according to (i) independent variables (main effects and interaction effects), (ii) data clustering (random effects), and (iii) data distribution (family and link functions). In the main effects tab, Tisane asks analysts if they would like to include additional or substitute main effects that Tisane infers to be conceptually relevant. In the interaction effects tab, Tisane suggests moderating relationships to include but does not automatically include them because analysts may not have specific hypotheses involving interactions (*DG3 - Guidance and control*). If analysts do not specify any moderating relationships, Tisane does not suggest any interaction effects, preventing analysts from including arbitrary interactions that may be conceptually unfounded (*DG1 - Conceptual knowledge*, *DG2 - Validity*).

In the data clustering tab, Tisane shows analysts which random effects it automatically includes based on the selected main and interaction effects. Unlike main and interaction effects, Tisane automatically includes random effects in order to maximize model generalizability (*DG2 - Validity*). If there is a random slope and random intercept pertaining to the same unit, Tisane asks analysts if they should be correlated or uncorrelated. We provide this option because analysts may have relevant domain expertise to make this decision (*DG3 - Guidance and control*). By default, Tisane correlates the random slope and random intercept.

The final tab, data distribution, helps analysts examine their data and select an initial family and link function to try. Appropriate selection of family and link functions depends on the data type of the dependent variable and the distribution of model residuals. Therefore, the selection can only be assessed after choosing a family and link function in the first place.

For an initial statistical model to consider, Tisane narrows the set of family functions considered based on the declared data type of variables (see 5.3.2) and lightweight viability checks, such as ensuring that a Poisson distribution is only applicable for variables that have nonnegative integer values. Tisane asks questions designed to uncover more semantically meaningful data types (e.g., counts) than are provided at variable declaration. Analysts without data can answer these questions as they are planning their studies (*DG4 - Statistical planning*). For the selected family candidate,

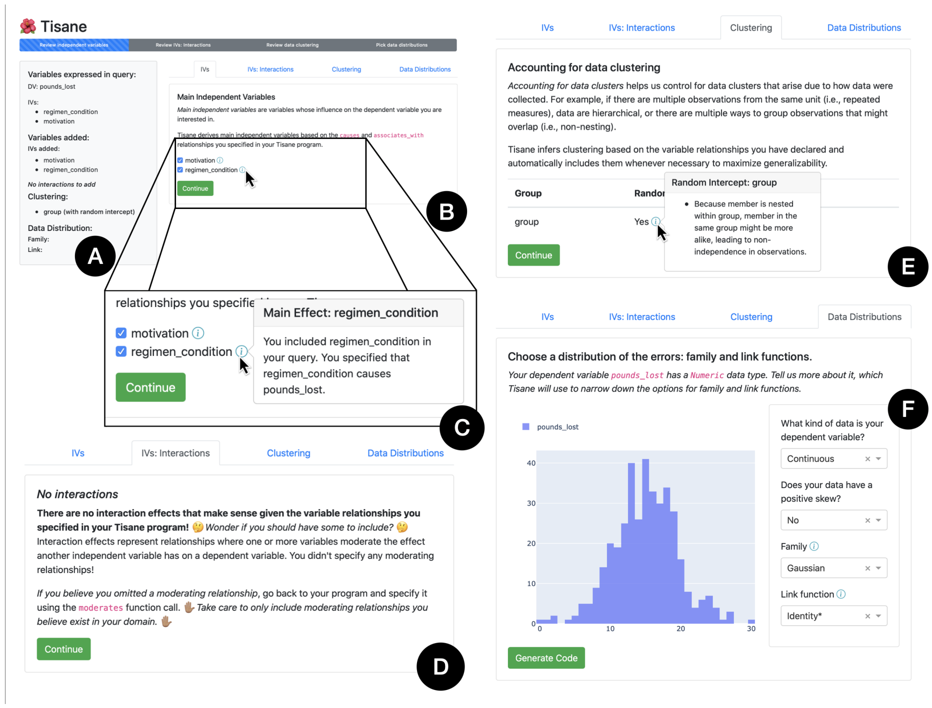


Figure 5.1: Example Tisane GUI for disambiguation. Tisane asks analysts disambiguating questions about variables that are conceptually relevant and that analysts may have overlooked in their query. (A) The left hand panel gives an overview of the model the analyst is constructing. (B) Based on the variable relationships analysts specify, Tisane infers candidate main effects that may be potential confounders. Tisane asks analysts if they would like to include these variables, explaining in a tooltip (C) why the variable may be important to include. (D) Tisane only suggests interaction effects if analysts specify moderating relationships in their specification. This way, Tisane ensures that model structures are conceptually justifiable. (E) From the data measurement relationships analysts provide, Tisane automatically infers and includes random effects to increase generalizability and external validity of statistical findings. (F) Tisane assists analysts in choosing an initial family and link function by asking them a series of questions about their dependent (e.g., Is the variable continuous or about count data?). To help analysts answer these questions and verify their assumptions about the data, Tisane shows a histogram of the dependent variable.

Tisane automatically selects the default link function based on the defaults for `statsmodels` Perktold et al. [2020] and `pymer4` Jolly [2018]. Analysts can then choose a different link function, as long as it is supported.

Output

There are two outputs of the interactive compilation: (ii) an executable modeling script and (ii) a log of GUI choices. To increase transparency of the authoring process, Tisane provides a log of user selections in the GUI as documentation, which the analyst can include in pre-registrations, for example (*DG4 - Statistical planning*). In the output script, Tisane includes code to fit the model and plot residuals against fitted values in order to assess the appropriateness of family and link functions, as is typical when examining family and link functions. The output script also includes a comment explaining what to look for in the plots and an online resource for further reading. Should analysts revise their choice of family and link functions, they can re-generate a script through the Tisane GUI.

5.4 Initial evaluation: Case studies with researchers

Given Tisane's novel focus on deriving and guiding analysts toward valid statistical models, we assessed how Tisane affects data analysis practices in three case studies with researchers. The following research questions guided the evaluation:

- **RQ1 - Workflow** How does Tisane's programming and interaction model affect how analysts author models? Specifically, what does Tisane make noticeably easier or more difficult when conducting an analysis?
- **RQ2 - Cognitive fixation** Where do researchers report spending more time or attention when using Tisane? How does this compare to their fixation during analyses typically?
- **RQ3 - Future possibilities** When do researchers imagine using Tisane in future projects, if at all? What additional support do researchers want from Tisane?

We recruited researchers through internal message boards and individual contacts. We intentionally recruited researchers at different stages of the research process—study planning, data analysis for publication, and ongoing model building and maintenance. We believed this could help us more holistically evaluate Tisane's impact on data analysis. We met with researchers over Zoom (R1, R3) and in person (R2) to discuss their use cases, observe them use Tisane for the first time, and ask for open-ended feedback. We pointed researchers to the Tisane tutorial for installation instructions and examples but otherwise encouraged the researchers to work independently. We answered any questions researchers had while using Tisane. Each study session lasted approximately 2 hours. At the end, two of the three researchers (R1, R3) said they planned to use Tisane again over the next two months.

5.4.1 Case Study 1: Planning a new study

R1, a clinical psychology PhD student, had recently submitted a paper and was planning a follow-up. R1 reported that she had never taken a formal class on modeling techniques but taught herself for her last paper. Her general workflow involved consulting with and mirroring what others in her research group did even if she did not completely understand why. R1 did not program often but said she had "enough coding experience to understand this kind of...[sample program]." Although familiar with Python, R1 preferred M+ Muthén [[n. d.]] and SPSS SPSS [2021]. She was interested in using Tisane to brainstorm new studies and research questions.

Using Tisane. After installation, R1 read through one of the computational notebook examples available in the Tisane GitHub repository. While reading, R1 asked clarifying questions about the variable types and syntax. R1 explained that the `Design` class felt novel because she had never seen the concept of a study design in data analysis code before. When the first two authors explained that it was supposed to be the equivalent of the statement of a study design in a paper, R1 remarked that usually, she "[kept] that in [her] head, which [she] probably shouldn't" (**RQ2 - Cognitive fixation**). Without a concrete data set, R1 preferred to walk through more examples rather than author a script of her own.

While reading an example, R1 drew a parallel between the tabs in SPSS dialogs for specifying models and the tabs in the Tisane GUI, noting that SPSS had a tab for control variables. R1 also wanted the ability to distinguish between "control variables" and other independent variables in the Tisane GUI. R1 explained that this would map more closely to how psychologists think about analyses. Future work could incorporate additional language constructs, such as a new data type for controls, for different groups of users (**RQ3 - Future possibilities**).

At the end of the study session, R1 remarked how Tisane "fills in a lot of the...gaps" in data analysis (**RQ1 - Workflow**, **RQ2 - Cognitive fixation**). The first gap R1 discussed was the *programming gap* between scientists and statistical tools. R1 believed that, for scientists who were not comfortable with programming, "they should probably be running less complex models, or first learn how to code" even if the complex models would be most appropriate. The second gap R1 discussed was the *statistical knowledge gap* in tools. R1 explained that in her experience, R provides support for more complex models but little guidance for what those models or statistical tests should be, requiring "top down assumption[s]." Thus, to R1, Tisane bridged the gap between tools like

SPSS and R by requiring minimal programming and providing modeling support. Put another way, Tisane bridged the gulf of execution Norman [2013] for R1 that previous tools had not.

5.4.2 Case Study 2: Analyzing data for a paper submission

R2, a computer science PhD student, had conducted a within-subjects study where 47 participants used four versions of an app for one week each (four weeks total). The motivating research question was how the different app designs led to psychological dissociation. Although R2 had expected to collect multiple survey responses for each participant each day, they only had aggregate daily self-report measures due to an error in the database management system. In the past, R2 reported having extensively explored their data and consulting others, but for this paper, they had not explored their data prior to fitting models because they felt more confident in their modeling skills. For analyses, R2 preferred R but had general Python programming experience. Prior to using Tisane, R2 had authored linear mixed effects models in R for their study. They were interested in using Tisane to check their analyses prior to submitting their paper to CHI.

Using Tisane. R2 wrote their scripts by adapting an example from the Tisane GitHub repository. As R2 considered which conceptual relationships to add, they reasoned aloud about if they should state causal or associative relationships between various measures and dissociation (**RQ2 - Cognitive fixation**). After some deliberation, they said, "I don't feel comfortable [making a causal statement]," and instead specified `associates_with` relationships. R1's hesitation to assert causal relationships confirms prior findings that specifying formal causal graphs is difficult for domain researchers Suzuki et al. [2020]; Suzuki and VanderWeele [2018]; Velentgas et al. [2013] and our design choice to allow for association edges. In addition, R2 was initially unsure about how to specify the `number_of_instances` for their measures since their original study design was unbalanced. After asking for clarification about `number_of_instances`, R2 declared all the measures with the parameter `number_of_instances` set equal to `date`.

Next, R2 ran their script and used the Tisane GUI in a browser window. Based on Tisane's recommended family and link functions, R2 realized the models they had previously authored in R using a Gaussian family were inappropriate. Due to a bug that we have since fixed, Tisane suggested a Poisson family that R2 used to generate a script, but this was an invalid choice given that not all dependent variable values were nonnegative integers. R2 explored other family distributions and generated a new script using an Inverse Gaussian family. When executed, the second output script issued an error due to the model inference algorithm failing to converge. R2 made a note to look into this model further on their own.

Once finished using Tisane, R2 commented that their analysis with Tisane was more streamlined (**RQ1 - Workflow**) in contrast to their very first paper where they had tried "every single kind of model that [they] could" until finding "the one that fits best," even if it was "one that no one would have heard of." R2 also stated they would be interested in using Tisane earlier in their analysis process in the future (**RQ3 - Future possibilities**). Based on their experience with Tisane, R2 questioned their previously authored linear mixed effects model, and said it was "unnerving" to discover an issue so close to a deadline. At the same time, they expressed, "if it's incorrect, I should know before I submit." A day after the study, R2 contacted the authors to inform them that they had decided to update their analyses from linear mixed effects models to generalized linear mixed effects models. They reported using the Inverse Gaussian family after visualizing and checking the distribution of residuals with help from the output Tisane script. The Inverse Gaussian family was appropriate because their dependent variable's values were all nonnegative and displayed a slight positive skew. R2's experience with Tisane suggests that Tisane can help researchers catch errors and lead them to re-examine their data, assumptions, and conclusions.

5.4.3 Case Study 3: Developing models to inform future models

Employed on a research team, R3 analyzes health data at the county, state, and national levels to estimate health expenditure and inform public policy. R3 develops initial models that are used to validate and generate estimates for larger, more comprehensive models. Due to the scale of data and established collaborative workflows, R3 typically works in a terminal or RStudio through a computing cluster and had very little experience with Python. Despite working on statistical models every day, R3 described himself as "not...a great modeler." R3 was interested in using Tisane to determine what variables to include as random effects in a model.

Using Tisane. R3 used Tisane in a local Jupyter notebook as well as on his team's cluster. R3 used the Tisane API overview reference material on GitHub to start writing his program, which involved copying and pasting the functions with their type signatures and then modifying them to match his dataset and incrementally running the program. The most common mistake R3 made while authoring his Tisane program was to refer to variables using the string names in the dataset (e.g., `"year"`) instead of the variable's alias (e.g., `year_id`), an idiom common in R but not in Python.

While authoring his Tisane program, R3 found the `number_of_instances` parameter redundant, especially because his data is always "square." Every `state_name` in his data set had 30 rows of data, corresponding to the `year_ids` 1990-2019. This is in contrast to R2, whose study design was unbalanced and resulted in variable numbers of observations per participant that needed to be aggregated. Based on R3's feedback, we added functionality to infer `number_of_instances` for each unit, which analysts can inspect by printing the variable.

While giving open-ended feedback on Tisane, R3, similar to R1, liked how Tisane helped "fill

[the] gap in...[his] knowledge" (**RQ2 - Cognitive fixation**). Given the diversity of models R3 works with, R3 found Tisane's focus on GLMs and GLMMs a "little limiting" and also wished to make Tisane "run without...the mouse" in a script, as is typical in his workflow (**RQ1 - Workflow**). Specifically, R3 described how he and his collaborators typically want to explore a space of models and run them in parallel. Nevertheless, R3 foresaw using Tisane in three types of modeling tasks common in his work: (i) exploratory modeling to determine if there are any interesting relationships between variables, (ii) authoring and comparing multiple models for prediction, and (iii) working out the precise model specification after identifying variables of interest (**RQ3 - Future possibilities**).

System changes and Takeaways

We fixed bugs and iterated on Tisane's GUI based on feedback from researchers. The largest change we made was to the data distributions tab. The data distributions tab we tested with researchers visualized the dependent variables against simulated distributions of family functions and included the results of the Shapiro-Wilk and D'Agostino and Pearson's normality tests. All three researchers reported becoming more aware of their data due to the visualizations. However, researchers' enthusiasm for the feature made us wary that visualizing the simulated data could mislead less careful analysts to believe that family and link functions pertain to variable distributions rather than the distributions of the model's residuals. To avoid such errors while still helping analysts become more aware of their data, we removed the simulated visualizations and normality tests and instead provide questions about the semantic nature of the dependent variable collected, as discussed in Section 5.3.2.

Overall, Tisane streamlines the analysis process (**RQ1 - Workflow**) in part because researchers report formalizing their conceptual knowledge into statistical models more directly (R1, R2). Although Tisane does not eliminate the need for model revision, Tisane may scope the revisions analysts consider to significant issues instead of details that may detract from the analysis goals (R2). Additionally, researchers reported a perceived shift in their attention from keeping track of and analyzing all possible modeling paths to their research questions and data assumptions (**RQ2 - Cognitive fixation**) while planning a new study and analysis (R1) as well as while preparing a research manuscript (R2). Future adoption of Tisane may depend on the complexity of analyses (**RQ3 - Future possibilities**) (R3). For instance, Tisane may provide a streamlined alternative to false starts due to misspecifications for simpler analyses (R1, R2, R3). For more complex models and studies, Tisane may act more as a prototyping tool for statistical models, helping researchers start at a reasonable model that they can then revise (R2, R3).

5.5 Limitations and Motivation for Re-design

Despite the importance of conceptual modeling to statistical analysis, it is unknown what challenges statistical novices face when expressing their domain knowledge. To identify what statistical non-experts want and are capable of expressing about their conceptual models, we conducted a formative study using Tisane **?**, an open-source tool designed to bridge this gap. We found important limitations to Tisane. Surprisingly, we found that some keywords and functions in Tisane were at too high a level of abstraction. Analysts wanted to express their conceptual relationships with greater detail and specificity. Analysts also wanted to express ambiguity about the a relationship's direction in the conceptual model.

VanderWeele [2019] (designed for scenarios where analysts are uncertain about the causal relationships in their domain). –> Cinelli et al.

5.6 Exploratory study

Reflection: As progress through PhD research, got and grappled with how to get closer to users and to statistical theory in tandem.

5.7 Second Release: rTisane

Update this section to match rTisane paper.

5.7.1 Updated DSL

5.7.2 Conceptual Model Disambiguation

5.7.3 Statistical model inference

5.8 Summative Evaluation: Lab study

Update this section to match rTisane paper.

5.9 Discussion: Do we want to a sep discussion for this chapter?

5.10 Summary of Contributions

Tisane embodies the hypothesis central to this dissertation: tool support for expressing implicit conceptual knowledge and reasoning about it to author statistical analyses enables statistical non-experts to specify valid statistical models. Through an iterative design process, we refined what the programming and interaction model for expressing conceptual models and connecting them to statistical models should be. Most notably, the second release of Tisane, as rTisane, provides more explicit support for conceptual model specification and disambiguation.

Tisane is a stark contrast to the current ecosystem of statistical analysis software designed to give analysts maximal mathematical and computational control at the cost of support for relating their statistical analyses with their conceptual meaning. The pending lab study results will demonstrate the impact of rTisane on (i) the conceptual models analysts specify and their reflection process, (ii) (output) statistical model quality, and (iii) awareness and learned insights analysts takeaway about their domain and data analysis process.

The first release of Tisane was a collaboration with Audrey Seo, Jeffrey Heer, and René Just. The corresponding paper was originally published and presented at ACM CHI 2022 cite, where it received a Best Paper Honorable Mention Award. The exploratory design study, second system iteration, and the summative evaluation are in collaboration with Edward Misback, Jeffrey Heer, and René Just. The second paper is under submission and has not yet been published.

Chapter 6

Conclusion

10 sprints: 10 minutes each –> 3 then stop/check in

While statistical analysis has become more pervasive among end-users who are not statistical experts, the tools for conducting analyses have continued to require high statistical expertise. This dissertation examines how to design and develop tools that not only lower the barriers for statistical non-experts but also provide guarantees about the validity of authored analyses. We introduce two new tools, Tea and Tisane. Chapter 3 introduced a DSL for Null Hypothesis Significance Testing. Chapter 5 introduced a DSL and interactive compilation process for authoring generalized linear models with or without mixed effects. Both are designed around the key insight that analysts <TODO: copy/paste from Pat email>. Analysts express their implicit knowledge about their domain and data–as assumptions and hypotheses in Tea and conceptual models in Tisane–and the DSLs compile them into statistical analysis code.

Additionally, we develop a theory of hypothesis formalization that describes the cognitive and operational steps involved in translating a conceptual research question into statistical analysis in code. Our theory of hypothesis formalization retrospectively validated our design in Tea, directly inspired the design of Tisane, unifies the formative observations that led to these system designs, and provides a framework for connecting and explaining the observations from our formative and summative studies.

A strength of this work is in how it integrates systems building with empirical studies, both of which motivated initial methodological experiments/innovations. We engaged in formative and summative evaluations to design, implement, iterate on, and evaluate Tea and Tisane. The evaluations involved qualitative and quantitative approaches. The empirical work helped us make technical insights in how to computationally represent statistical analysis–as constraints and as graphs. Furthermore, we took a deep dive into triangulating the nuances involved in authoring statistical analyses through a qualitative analysis, lab study, and tools assessment in order to develop our theory of hypothesis formalization.

In addition to the formal empirical studies we conducted in this dissertation, we benefited from informal observations, reports from early users, and our personal experiences throughout the design processes. For instance, Tea came from years of personal experiences and informal observations of how computer scientists author statistical analyses relying, at best, on charts and tables describing when specific tests were applicable.

6.1 Impact

Fill in

6.2 Discussion: Themes

6.2.1 Designing the *right* levels of abstraction (Challenge 1)

With any domain-specific language of software system, there is a formalism end-users have to learn, where does it come from, how well does it align with what they want to do/say, etc.

Both Tea and Tisane provide higher levels of abstraction for analysts to express their implicit assumptions to author statistical analyses. However, the key to the systems was not that they were just higher level but rather that their abstractions were at the appropriate conceptual and data collection details that analysts could specify and was still amenable to rigorous reasoning. In fact, a key insight that guided our re-design of Tisane (see Section 5.6) was that analysts wanted low- and high-level conceptual abstractions. Therefore, a key takeaway from this work is that higher levels of abstraction are not always better. Rather, abstractions that allow analysts to dig deeper into the parts they want to and can is what is necessary and impactful.

Furthermore, the conceptual relationships between variables were still implicit in Tea, but we made them more prominent primitives in Tisane. Our work on defining hypothesis formalization helped us to identify the centrality/importance of grappling with conceptual relationships explicitly.

This focus on the conceptual knowledge analysts can express and can guide computational and statistical reasoning highlights/suggests a shift in perspective our perspective on the design problem at hand with statistical analysis. While much effort has been put toward making statistical computation more precise and efficient and the mathematical abstractions expressive, the real design barrier lies in the conceptualization of the problem of statistical analysis. That is, statistical analysis is a means to an end for many analysts, especially statistical non-experts. Analysts' primary goal is to understand something about their domain. Therefore, statistical software should serve this goal, by allowing analysts to think about their domains and goals for analysis deeply while authoring analyses (e.g., by documenting their implicit assumptions about their domains) and interpret the results of the analyses in light of their conceptual domain knowledge. This view aligns with a familiar breakdown of complex tasks into the gulfs of execution and evaluation, respectively. While this thesis has focused on how to bridge the gulf of execution, there is much important work on how to report and help analysts interpret the results of their analyses in light of their conceptual assumptions and models.

6.2.2 Balancing user and computational reasoning / representations for automated reasoning (Challenge 2)

A core challenge in Tea and Tisane was in designing "shared representations" between users and computational techniques that could enable formal reasoning about analyses. A key insight in Tea was that statistical test selection could be reformulated as a constraint satisfaction problem. As a result, we constructed a knowledge base representing statistical tests using logical constraints. Using Tea's DSL, analysts specify additional constraints about their hypothesis and data. Tea reasons over these constraints to identify valid statistical tests, which it then executes. In Tisane, the shared representation is a graph, which contains a causal subgraph useful for deriving linear models.

In designing these representations, a temptation was to fit the DSL on top of a reasoning approach that was straightforward. In this view, the DSL was a thin wrapper around the automated reasoning engine. For example, a very early prototype of Tisane used Answer Set Programming (ASP) to define when specific confounders should appear in a generalized linear model. In addition to being a clunky way to represent linear model formulation rules when the statistics community has converged on using graphs, this prototype required analysts to incrementally refine their specification by interacting with the UNSAT core. Although an interesting, informative prototype, this under-designed the disambiguation to reap its benefits. Interaction is not just for getting the system to find an answer but a way for users to be able to not only incrementally express their intents for analysis but also reflect and refine their understanding of the domain and data. In other words, finding and using shared representations require designing not only the programming abstractions but also the interactions with the abstractions.

6.2.3 The role of programs and the act of programming as a reflective practice (Challenge 3)

Initially, we were surprised when analysts using Tea reported finding it useful to learn about their analyses....

Providing abstractions that prioritize the knowledge and motivations end-users have in authoring statistical analyses promotes reflection, making this a more reflective practice (Schon). This work asks how we can make not only the programs themselves useful for accomplishing a task (i.e., running a statistical test or model) but also the programs themselves useful for documentation and sharing as well as the process of programming reflective and meaningful to end-users. The implicit assumptions analysts have about their data are explicitly stated in a Tea program. In Tisane, the assumed conceptual relationships are also encoded and captured. These are useful for planning, sharing, and being accountable in the future, as seen in pre-registration system using Tea **?**. In this way, a central theme that this work addresses is how to reify the connection between the conceptual and statistical in our software tools. In this way, this dissertation brings to the domain of data analysis, classic principles from end-user software engineering

6.3 Recent developments

Do this during revisions?

6.3.1 Construct validity: Within reach with the usage of LLMs

This thesis focused on internal, external, and statistical conclusion validity. However, could reason about construct validity with LLMs.

6.3.2 What about in the face of LLMs?

But how do people express their domain knowledge, make the process meaningful

Mention LLMs as a technology to use here?

6.4 Limitations and Future work

Some themes that emerge: - Support for after statistical modeling – what do these mean? (interpretation) –> What to do next? / what would help me answer my research question? (modeling-testing) –> - How to capture and use these consequences into next phases of the lifecycle? - How could all these improve science and make data analysis more robust?

By addressing the important trend of the increased diversity of end-users authoring statistical analyses and the importance of correct analyses, this dissertation opens up statistical analysis authoring as a domain for further research.

This dissertation dissects statistical data analysis from an activity taken for granted into a process fraught with problems for end-users that have high impact consequences. Improving statistical analysis authoring opens up questions in addressing core issues in statistics, end-user software engineering, programming language design, which we hope will be additional directions other pursue.

Here we elaborate one some of the limitations of this work and opportunities for future research in each of these disciplines.

6.4.1 Connecting modeling with testing

A natural question to ask at the end of this thesis is, "Which system should an analyst use? Tea or Tisane?" Tea and Tisane serve different statistical purposes. Tea is focused on statistical testing, or finding if there is evidence in the data for or against a specific claim, while Tisane is focused on statistical modeling, trying to estimate the influence of a variable (or sets of variables) on another variable, given the messy nature of the world, including confounding, mediation, moderation, etc. Statistical testing and modeling are not mutually exclusive. In fact, statistical experts often perform statistical tests after building statistical models. Mathematically, all statistical tests can be reformulated into statistical models with specific parameters of interest serving as test statistics (see for an approachable summary).

What we have observed in our studies and personal experiences is that analysts often reach for statistical tests even when what they really need is a statistical model. Analysts will even contort their research questions to fit the (sub-optimal) statistical tests they can implement, just as we saw in hypothesis formalization.

Tea and Tisane do not address a key limitation of the current ecosystem of statistical software, which is guidance in what analysis approach to take. A compelling next step in this work is to allow analysts to ask follow-up queries after authoring a statistical model to probe into what its implications are and test the differences between groups given a model. To make this possible, additional querying and disambiguation after outputting a statistical model from Tisane are necessary. What would make this difficult is...

6.4.2 Interpretation of results

While Tisane addresses the gulf of execution in authoring statistical analyses to answer a research question, it falls short of addressing the gulf of evaluation. Tisane does not yet help analysts interpret the results of their statistical models in light of their expressed implicit conceptual knowledge. For scientific discovery and decision making, accurately interpreting statistical results is equally important. For example, if an analysts' statistical results suggest that there is no evidence in the data to support the existence of a relationship in their conceptual model, how does the analyst make this interpretation? What should the analyst do about it? Is there conceptual model "incorrect"? Should they revise their conceptual model? Check their data collection procedure? To answer these questions, there are two related challenges to address: (i) improved statistical reporting (What do the results mean?) and (ii) support for navigating consequences, such as through richer or follow-up queries and model revisions (What should an analyst do next?).

Furthermore, in the long-term, to support more complex analyses, there is a need to support more types of analyses.

Tea does this a bit better.

6.4.3 Support throughout the data lifecycle

This dissertation identifies the need for improved abstractions for authoring statistical analyses. I argue that the appropriate abstractions should capture the implicit domain knowledge analysts bring to their data and show the benefits to users for doing so: valid statistical analysis formulation and increased reflection among analysts about their domain and data. From an engineering perspective, these abstractions allow tool designers to separate the conceptual and statistical concerns involved in data analysis, using Tea and Tisane as platforms for experimenting with alternative statistical model derivations and formulations. Given that implicit assumptions about a domain pervade the entire data lifecycle, what if we could take a similar approach throughout? What would the appropriate data structures for capturing domain knowledge look like at each phase? How could a new ecosystem of software tools use these representations such that the evolution of conceptual knowledge could be tracked and traced in meaningful ways. – some research questions to ask/address

A fertile area to try this integration is in connecting statistical analysis with visual analysis

6.4.4 Improving science

The systems in this dissertation show a way to author valid statistical analyses by design. Tea incorporates formal methods to statistical test selection, and Tisane incorporates causal reasoning into model authoring. These efforts are in contrast to existing systems that place the burden of validity on end-users. These systems are a step from threats to validity to guarantees about validity.

To truly improve the quality and reliability of science, a more end-to-end approach, even before statistical authoring, is important to design for. Recently, researchers have used Tea to support study planning and pre-registration **?**. However, support for identifying interesting research questions and hypotheses in addition to planning experiments and data collection methods, would help. This is possible given that we can treat these reasoning engines to reason in multiple directions, from hypotheses to statistical models or statistical models to hypotheses, research questions, and assumptions about data. Could even incorporate some data insights (e.g., structure learning). Tea and Tisane focus on a top-down authoring approach where analysts start with a research question and hypothesis. However, as we saw in hypothesis formalization, analysts may refine their hypotheses in response to statistical results. Therefore, incorporating both data-driven and research question-driven approaches to model authoring and refinement will be an important next step. To do so, need to support interpretation and refinement.

One of the precautions we designed Tisane around was preventing cherry-picking and p-hacking by involving analysts in the statistical model formulation process. Tisane supports one conceptual model to a statistical model. However, to assess robustness of an effect, across multiple possible conceptual models or where there is ambiguity, analysts might need to consider multiple possible conceptual models. We did this by having analysts pick one final model, but there could be ways to further embrace the uncertainty in conceptual models and statistical model formulations by authoring a multiverse of statistical models to measure the robustness of statistical results. By reporting out the sensitivity of a result, could address cherry-picking concerns.

An interesting observation we made in our lab study to develop hypothesis formalization was an insistence on being "data-driven," which meant refusing to state implicit assumptions explicitly.

Formal methods for science E.g., github for scientific checking

6.4.5 Methods for human-centered programming language design

From Tea to Tisane, we changed how we designed DSL primitives. To identify primitives in Tea's DSL, we surveyed two introductory quantitative methods courses in human-computer interaction. For Tisane, we started by iterating on primitives that made sense to us, as designers, and would be amenable to formal reasoning. However, between the first and second releases, we sought to further incorporate what analysts want to express to increase the likelihood of them using the system correctly. In this process, we sought to strike the right balance between designer and end-user participatory design. While general approaches for "end-user programming" have been developed, such as in PLIERS , there are not yet methods for how to adapt DSLs over time. Recent work on Stitch finds ways to improve DSLs through data on API usage. This is a promising direction, and earlier ways to prototype APIs with end-users with a similar flavor could be helpful.

Methods for specializing DSLs are also promising/important.

need for human-centered methods to design DSLs

6.5 Closing Remarks

Fill in after intro

Bibliography

Sara Alspaugh, Nava Zokaei, Andrea Liu, Cindy Jin, and Marti A Hearst. 2018. Futzing and moseying: Interviews with professional data analysts on exploration practices. *IEEE transactions on visualization and computer graphics* 25, 1 (2018), 22–31.

American Psychological Association et al. 1983. *Publication manual.* American Psychological Association Washington, DC.

Eytan Bakshy, Dean Eckles, and Michael S Bernstein. 2014. Designing and deploying online field experiments. In *Proceedings of the 23rd international conference on World wide web.* ACM, 283–292.

Dale J Barr. 2013. Random effects structure for testing interactions in linear mixed-effects models. *Frontiers in psychology* 4 (2013), 328.

Dale J Barr, Roger Levy, Christoph Scheepers, and Harry J Tily. 2013. Random effects structure for confirmatory hypothesis testing: Keep it maximal. *Journal of memory and language* 68, 3 (2013), 255–278.

Douglas Bates, Martin Mächler, Ben Bolker, and Steve Walker. 2014. Fitting linear mixed-effects models using lme4. *arXiv preprint arXiv:1406.5823* (2014).

Douglas Bates, Martin Mächler, Ben Bolker, Steve Walker, Rune H.B. Christensen, Henrik Singmann, Bin Dai, Fabian Scheipl, Gabor Grothendieck, Peter Green, and John Fox. 2019. Package 'lme4'. *CRAN* (2019). "https://cran.r-project.org/web/packages/lme4/lme4.pdf"

Leilani Battle and Jeffrey Heer. 2019. Characterizing Exploratory Visual Analysis: A Literature Review and Evaluation of Analytic Provenance in Tableau. *Computer Graphics Forum (Proc. EuroVis)* (2019). "http://idl.cs.washington.edu/papers/exploratory-visual-analysis"

Jacques Bertin. 2011. *Graphics and graphic information processing.* Walter de Gruyter.

Graeme Blair, Jasper Cooper, Alexander Coppock, and Macartan Humphreys. 2019. Declaring and diagnosing research designs. *American Political Science Review* 113, 3 (2019), 838–859.

Igor Bobriakov. 2017. Top 15 Python Libraries for Data Science in 2017. *ActiveWizards in Medium* (2017). "https://medium.com/activewizards-machine-learning-company/top-15-python-libraries-for-data-science-in-in-2017-ab61b4f9b4a7"

Igor Bobriakov. 2018. Top 20 Python libraries for data science in 2018. *ActiveWizards in Medium* (2018). "https://medium.com/activewizards-machine-learning-company/top-20-python-libraries-for-data-science-in-2018-2ae7d1db8049"

Benjamin M Bolker, Mollie E Brooks, Connie J Clark, Shane W Geange, John R Poulsen, M Henry H Stevens, and Jada-Simone S White. 2009. Generalized linear mixed models: a practical guide for ecology and evolution. *Trends in ecology & evolution* 24, 3 (2009), 127–135.

Leo Breiman, Adele Cutler, Andy Liaw, and Matthew Wiener. 2018. Package 'randomForest'. (2018). "https://cran.r-project.org/web/packages/randomForest/randomForest.pdf"

Mollie E Brooks, Kasper Kristensen, Koen J van Benthem, Arni Magnusson, Casper W Berg, Anders Nielsen, Hans J Skaug, Martin Machler, and Benjamin M Bolker. 2017. glmmTMB balances speed and flexibility among packages for zero-inflated generalized linear mixed modeling. *The R journal* 9, 2 (2017), 378–400.

J. Bruin. 2019. Choosing the Correct Statistical Test in SAS, Stata, SPSS and R. https://stats.idre.ucla.edu/other/mult-pkg/whatstat/

Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, et al. 2013. API design for machine learning software: experiences from the scikit-learn project. *arXiv preprint arXiv:1309.0238* (2013).

Paul-Christian Bürkner et al. 2017. brms: An R package for Bayesian multilevel models using Stan. *Journal of statistical software* 80, 1 (2017), 1–28.

Paul-Christian Bürkner and Maintainer Paul-Christian Buerkner. 2016. Package 'brms'. (2016).

Bob Carpenter, Andrew Gelman, Matthew D. Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. 2017. Stan : A Probabilistic Programming Language. *Journal of Statistical Software* 76 (01 2017). `https://doi.org/10.18637/jss.v076.i01`

Robert Carver, Michelle Everson, John Gabrosek, Nicholas Horton, Robin Lock, Megan Mocko, Allan Rossman, Ginger Holmes Roswell, Paul Velleman, Jeffrey Witmer, et al. 2016. Guidelines for assessment and instruction in statistics education (GAISE) college report 2016. (2016).

Yunshun Chen, Aaron TL Lun, Davis J McCarthy, Matthew E Ritchie, Belinda Phipson, Yifang Hu, Xiaobei Zhou, Mark D Robinson, and Gordon K Smyth. 2020a. Empirical Analysis of Digital Gene Expression Data in R (v3.30.3). (2020). `"https://bioconductor.org/packages/release/bioc/html/edgeR.html"`

Yunshun Chen, David McCarthy, Matthew Ritchie, Mark Robinson, and Gordon Smyth. 2020b. edgeR: differential analysisof sequence read count data. (2020). `"https://bioconductor.org/packages/release/bioc/vignettes/edgeR/inst/doc/edgeRUsersGuide.pdf"`

François Chollet et al. 2015. Keras. `https://keras.io`.

Herbert H Clark. 1973. The language-as-fixed-effect fallacy: A critique of language statistics in psychological research. *Journal of verbal learning and verbal behavior* 12, 4 (1973), 335–359.

Jacob Cohen. 1988. Statistical power analysis for the social sciences. (1988).

Jacob Cohen, Patricia Cohen, Stephen G West, and Leona S Aiken. 2013. *Applied multiple regression/correlation analysis for the behavioral sciences.* Routledge.

Plotly Dash Community. [n. d.]. Plotly Dash. `https://plotly.com/dash/`

Joshua R De Leeuw. 2015. jsPsych: A JavaScript library for creating behavioral experiments in a Web browser. *Behavior research methods* 47, 1 (2015), 1–12.

Leonardo De Moura and Nikolaj Bjørner. 2008. Z3: An efficient SMT solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 337–340.

Bradley Efron. 1992. Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics*. Springer, 569–593.

Isaac Ehrlich. 1973. Participation in illegitimate activities: A theoretical and empirical investigation. *Journal of political Economy* 81, 3 (1973), 521–565.

Alexander Eiselmayer, Chatchavan Wacharamanotham, Michel Beaudouin-Lafon, and Wendy Mackay. 2019. Touchstone2: An Interactive Environment for Exploring Trade-offs in HCI Experiment Design. (2019).

Janet Feigenspan, Christian Kästner, Jörg Liebig, Sven Apel, and Stefan Hanenberg. 2012. Measuring programming experience. In *2012 20th IEEE International Conference on Program Comprehension (ICPC)*. IEEE, 73–82.

Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. 2015. Efficient and Robust Automated Machine Learning. In *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (Eds.), Vol. 28. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2015/file/11d0e6287202fced83f79975ec59a3a6-Paper.pdf

Andy Field, Jeremy Miles, and Zoë Field. 2012. *Discovering statistics using R.* Sage publications.

Ronald Aylmer Fisher. 1937. *The design of experiments.* Oliver And Boyd; Edinburgh; London.

Jerome Friedman, Trevor Hastie, Rob Tibshirani, Balasubramanian Narasimhan, Kenneth Tay, Noah Simon, and Junyang Qian. 2020. Package 'glmnet'. (2020). "https://cran.r-project.org/web/packages/glmnet/index.html"

Jonah Gabry, Daniel Simpson, Aki Vehtari, Michael Betancourt, and Andrew Gelman. 2019. Visualization in Bayesian workflow. *Journal of the Royal Statistical Society: Series A (Statistics in Society)* 182, 2 (2019), 389–402.

Andrew Gelman. 2005. Why I don't use the term "fixed and random effects". https://statmodeling.stat.columbia.edu/2005/01/25/why_i_dont_use/

Andrew Gelman and Jennifer Hill. 2006. *Data analysis using regression and multilevel/hierarchical models.* Cambridge university press.

Andrew Gelman and Eric Loken. 2013. The garden of forking paths: Why multiple comparisons can be a problem, even when there is no "fishing expedition" or "p-hacking" and the research hypothesis was posited ahead of time. *Department of Statistics, Columbia University* (2013).

LLC. GraphPad Software. 2020. GraphPad Prism 8 User Guide. (2020). "https://www.graphpad.com/guides/prism/8/user-guide/index.htm"

Sander Greenland, Judea Pearl, and James M Robins. 1999. Causal diagrams for epidemiologic research. *Epidemiology* (1999), 37–48.

Garrett Grolemund. 2019. Quick list of useful R packages. *R Studio Support* (2019). `"https://support.rstudio.com/hc/en-us/articles/201057987-Quick-list-of-useful-R-packages"`

Garrett Grolemund and Hadley Wickham. 2014. A cognitive interpretation of data analysis. *International Statistical Review* 82, 2 (2014), 184–204.

François Guimbretière, Morgan Dixon, and Ken Hinckley. 2007. ExperiScope: an analysis tool for interaction data. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 1333–1342.

Jarrod Hadfield. 2020. Package ‘MCMCglmm‘. (2020). `"https://cran.r-project.org/web/packages/MCMCglmm/MCMCglmm.pdf"`

Jarrod D Hadfield et al. 2010. MCMC methods for multi-response generalized linear mixed models: the MCMCglmm R package. *Journal of Statistical Software* 33, 2 (2010), 1–22.

Trevor Hastie and Junyang Qian. 2014. Glmnet vignette. (2014). `https://web.stanford.edu/~hastie/glmnet/glmnet_alpha.html`

Brian Hempel, Justin Lubin, and Ravi Chugh. 2019. Sketch-n-Sketch: Output-Directed Programming for SVG. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 281–292.

Jane Hoffswell, Alan Borning, and Jeffrey Heer. 2018. SetCoLa: High-Level Constraints for Graph Layout. In *Computer Graphics Forum*, Vol. 37. Wiley Online Library, 537–548.

John H Holland, Keith J Holyoak, Richard E Nisbett, and Paul R Thagard. 1989. *Induction: Processes of inference, learning, and discovery.* MIT press.

Sture Holm. 1979. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics* (1979), 65–70.

Rick H Hoyle. 1995. *Structural equation modeling: Concepts, issues, and applications.* Sage.

SAS Institute Inc. 2021. SAS. `https://www.sas.com/`

Eshin Jolly. 2018. Pymer4: connecting R and Python for linear mixed modeling. *Journal of Open Source Software* 3, 31 (2018), 862.

Bradley Jones and John Sall. 2011. JMP statistical discovery software. *Wiley Interdisciplinary Reviews: Computational Statistics* 3, 3 (2011), 188–194.

Eric Jones, Travis Oliphant, Pearu Peterson, et al. 2001–2021a. SciPy: Open source scientific tools for Python. `http://www.scipy.org/`

Eric Jones, Travis Oliphant, Pearu Peterson, et al. 2001–2021b. SciPy: Open source scientific tools for Python. `https://docs.scipy.org/doc/scipy/reference/stats.html`

Eric Jones, Travis Oliphant, Pearu Peterson, et al. 2001–2021c. SciPy: Open source scientific tools for Python. `https://docs.scipy.org/doc/scipy/reference/optimize.html`

Eunice Jun, Melissa Birchfield, Nicole De Moura, Jeffrey Heer, and Rene Just. 2022. Hypothesis Formalization: Empirical Findings, Software Limitations, and Design Implications. In *ACM Transactions on Computer-Human Interaction (TOCHI)*, Vol. 29. Issue 1. `"https://arxiv.org/abs/2104.02712"`

Eunice Jun, Maureen Daum, Jared Roesch, Sarah E Chasins, Emery D Berger, Rene Just, and Katharina Reinecke. 2019. Tea: A High-level Language and Runtime System for Automating Statistical Analysis. In *Proceedings of the 32nd Annual Symposium on User Interface Software and Technology*. ACM.

Robert I Kabacoff. 2011. R: In Action. (2011).

Alex Kale, Matthew Kay, and Jessica Hullman. 2019. Decision-making under uncertainty in research synthesis: Designing for the garden of forking paths. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–14.

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.

Sean Kandel, Andreas Paepcke, Joseph M Hellerstein, and Jeffrey Heer. 2012. Enterprise data analysis and visualization: An interview study. *IEEE Transactions on Visualization and Computer Graphics* 18, 12 (2012), 2917–2926.

Maurits Kaptein and Judy Robertson. 2012. Rethinking statistical analysis methods for CHI. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1105–1114.

Matthew Kay, Gregory L Nelson, and Eric B Hekler. 2016. Researcher-centered design of statistics: Why Bayesian statistics better fit the culture and incentives of HCI. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 4521–4532.

Moe Kayali, Babak Salimi, and Dan Suciu. 2020. Demonstration of inferring causality from relational databases with CaRL. *Proceedings of the VLDB Endowment* 13, 12 (2020), 2985–2988.

Norbert L Kerr. 1998. HARKing: Hypothesizing after the results are known. *Personality and social psychology review* 2, 3 (1998), 196–217.

David Klahr and Kevin Dunbar. 1988. Dual space search during scientific reasoning. *Cognitive science* 12, 1 (1988), 1–48.

David Klahr and Herbert A Simon. 1999. Studies of scientific discovery: Complementary approaches and convergent findings. *Psychological Bulletin* 125, 5 (1999), 524.

Gary Klein, Jennifer K Phillips, Erica L Rall, and Deborah A Peluso. 2007. A data–frame theory of sensemaking. In *Expertise out of context*. Psychology Press, 118–160.

Scott Klemmer and Jacob Wobbrock. 2019. Designing, Running, and Analyzing Experiments. https://www.coursera.org/learn/designexperiments

Joos Korstanje. 2019. "ANOVA's three types of estimating Sums of Squares: don't make the wrong choice!". *Towards Data Science, Medium* (2019). "https://towardsdatascience.com/anovas-three-types-of-estimating-sums-of-squares-don-t-make-the-wrong-choice-91107c77a27a"

Ita GG Kreft, Ita Kreft, and Jan de Leeuw. 1998. *Introducing multilevel modeling*. Sage.

Max Kuhn, Davis Vaughan, and RStudio. 2020. parsnip: A Common API to Modeling and Analysis Functions. "https://parsnip.tidymodels.org/"

Max Kuhn and Hadley Wickham. 2020. *Tidymodels: a collection of packages for modeling and machine learning using tidyverse principles.* "https://www.tidymodels.org"

Erin LeDell and Sebastien Poirier. 2020. H2O AutoML: Scalable Automatic Machine Learning. *7th ICML Workshop on Automated Machine Learning (AutoML)* (July 2020). https://www.automl.org/wp-content/uploads/2020/07/AutoML_2020_paper_61.pdf

Kent State University Libraries. 2019. SPSS Tutorials: Analyzing Data. https://libguides.library.kent.edu/SPSS/AnalyzeData

Jiali Liu, Nadia Boukhelifa, and James R Eagan. 2019b. Understanding the Role of Alternatives in Data Analysis Practices. *IEEE transactions on visualization and computer graphics* 26, 1 (2019), 66–76.

Yang Liu, Tim Althoff, and Jeffrey Heer. 2019a. Paths Explored, Paths Omitted, Paths Obscured: Decision Points & Selective Reporting in End-to-End Data Analysis. *arXiv preprint arXiv:1910.13602* (2019).

StataCorp LLC. 2020a. Language syntax. `https://www.stata.com/manuals13/u11.pdf`

StataCorp LLC. 2020b. Stata 16 Documentation. `https://www.stata.com/features/documentation/`

James Lloyd, David Duvenaud, Roger Grosse, Joshua Tenenbaum, and Zoubin Ghahramani. 2014. Automatic construction and natural-language description of nonparametric regression models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 28.

Steson Lo and Sally Andrews. 2015. To transform or not to transform: Using generalized linear mixed models to analyse reaction time data. *Frontiers in psychology* 6 (2015), 1171.

Calvin Loncaric, Emina Torlak, and Michael D Ernst. 2016. Fast synthesis of fast collections. *ACM SIGPLAN Notices* 51, 6 (2016), 355–368.

David J. Lunn, Andrew Thomas, Nicky Best, and David Spiegelhalter. 2000. WinBUGS - A Bayesian modelling framework: Concepts, structure, and extensibility. *Statistics and Computing* 10, 4 (01 Oct 2000), 325–337. `https://doi.org/10.1023/A:1008929526011`

Wendy E Mackay, Caroline Appert, Michel Beaudouin-Lafon, Olivier Chapuis, Yangzhou Du, Jean-Daniel Fekete, and Yves Guiard. 2007. Touchstone: exploratory design of experiments. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 1425–1434.

Arni Magnusson, Hans Skaug, Anders Nielsen, Casper Berg, Kasper Kristensen, Martin Maechler, Koen van Bentham, Ben Bolker, Nafis Sadat, Daniel Lüdecke, Russ Lenth, Joseph O'Brien, and Mollie Brooks. 2020. Package 'glmmTMB'. (2020). `"https://cran.r-project.org/web/packages/glmmTMB/index.html"`

Richard McElreath. 2020. *Statistical rethinking: A Bayesian course with examples in R and Stan.* CRC press.

Dominik Moritz, Chenglong Wang, Greg L Nelson, Halden Lin, Adam M Smith, Bill Howe, and Jeffrey Heer. 2019. Formalizing visualization design knowledge as constraints: Actionable and extensible models in Draco. *IEEE transactions on visualization and computer graphics* 25, 1 (2019), 438–448.

Muthén & Muthén. [n. d.]. MPlus. `https://www.statmodel.com/`

John Ashworth Nelder and Robert WM Wedderburn. 1972. Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)* 135, 3 (1972), 370–384.

Allen Newell, Herbert Alexander Simon, et al. 1972. *Human problem solving*. Vol. 104. Prentice-Hall Englewood Cliffs, NJ.

Don Norman. 2013. *The design of everyday things: Revised and expanded edition*. Basic books.

Donald A Norman. 1986. Cognitive engineering. *User centered system design* 31 (1986), 61.

University of Amsterdam. 2020. JASP: A Fresh Way to do Statistics. `"https://jasp-stats.org/"`

Travis E Oliphant. 2006. *A guide to NumPy*. Vol. 1. Trelgol Publishing USA.

Pavel Panchekha, Adam T Geller, Michael D Ernst, Zachary Tatlock, and Shoaib Kamil. 2018. Verifying that web pages have accessible layout. In *Proceedings of the 39th ACM SIGPLAN Conference on Programming Language Design and Implementation*. ACM, 1–14.

Judea Pearl. 1995a. Causal diagrams for empirical research. *Biometrika* 82, 4 (1995), 669–688.

Judea Pearl. 1995b. Causal diagrams for empirical research. *Biometrika* 82, 4 (1995), 669–688.

Judea Pearl et al. 2000. Models, reasoning and inference. *Cambridge, UK: CambridgeUniversityPress* 19 (2000).

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* 12 (2011), 2825–2830.

Josef Perktold, Skipper Seabold, Jonathan Taylor, and statsmodels developers. 2020. Statsmodels v0.10.2 Reference Guide. (2020). `"https://www.statsmodels.org/stable"`

M Pfannkuch. 1997. Statistical thinking: One statistician's perspective. *Research papers on stochastics education* (1997), 171–178.

Maxine Pfannkuch, Chris J Wild, et al. 2000. Statistical Thinking an Statistical Practice: Themes Gleaned from Professional Statisticians. *Statistical science* 15, 2 (2000), 132–152.

José Pinheiro, Douglas Bates, Saikat DebRoy, Deepayan Sarkar, EISPACK authors, Siem Heisterkamp, Bert Van Willigen, and R-core. 2020. Package ‘nlme‘. (2020). "https://cran.r-project.org/web/packages/nlme/nlme.pdf"

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. McLean, VA, USA, 2–4.

Tanu N Prabhu. 2019. Top Python Libraries Used In Data Science. *Towards Data Science, Medium* (2019). "https://towardsdatascience.com/top-python-libraries-used-in-data-science-a58e90f1b4ba"

Brian Ripley, Bill Venables, Douglas M. Bates, Kurt Hornik, Albrecht Gebhardt, , and David Firth. 2020. Package ‘MASS‘. (2020). "https://cran.r-project.org/web/packages/MASS/MASS.pdf"

Donald B Rubin. 2004. Teaching statistical inference for causal effects in experiments and observational studies. *Journal of Educational and Behavioral Statistics* 29, 3 (2004), 343–367.

Daniel M Russell, Mark J Stefik, Peter Pirolli, and Stuart K Card. 1993. The cost structure of sensemaking. In *Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems*. ACM, 269–276.

Babak Salimi, Harsh Parikh, Moe Kayali, Lise Getoor, Sudeepa Roy, and Dan Suciu. 2020. Causal relational learning. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 241–256.

Michel F Sanner et al. 1999. Python: a programming language for software integration and development. *J Mol Graph Model* 17, 1 (1999), 57–61.

SAS. 2020. JMP. "https://www.jmp.com/en_us/home.html"

Arvind Satyanarayan, Dominik Moritz, Kanit Wongsuphasawat, and Jeffrey Heer. 2017. Vega-lite: A grammar of interactive graphics. *IEEE transactions on visualization and computer graphics* 23, 1 (2017), 341–350.

Christian D Schunn and David Klahr. 1995. A 4-space model of scientific discovery. In *Proceedings of the 17th annual conference of the cognitive science society*. 106–111.

Christian D Schunn and David Klahr. 1996. When and how to go beyond a 2-space model of scientific discovery. In *Proceedings of the Eighteenth Annual Conference of the Cognitive Science Society: July 12-15, 1996, University of California, San Diego*, Vol. 18. Psychology Press, 25.

scikit-learn developers. 2020. Scikit-Learn v0.23.2 Documentation. `https://scikit-learn.org/stable/`

Skipper Seabold and Josef Perktold. 2010. Statsmodels: Econometric and statistical modeling with python. In *Proceedings of the 9th Python in Science Conference*, Vol. 57. Scipy, 61.

William R Shadish. 2010. Campbell and Rubin: A primer and comparison of their approaches to causal inference in field settings. *Psychological methods* 15, 1 (2010), 3.

N.J.A. Sloane and R.H. Hardin. 2017. Gosset: A General-purpose program for designing experiments. `http://neilsloane.com/gosset/`

Peter Spirtes. 1994. Conditional independence in directed cyclic graphical models for feedback. (1994).

Peter Spirtes, Thomas Richardson, Christopher Meek, Richard Scheines, and Clark Glymour. 1996. Using d-separation to calculate zero partial correlations in linear models with correlated errors. *Publisher: Carnegie Mellon University* (1996).

IBM SPSS. 2021. SPSS Software. `https://www.ibm.com/analytics/spss-statistics-software`

Stata. 2021. Stata Software. `https://www.stata.com/`

Michael Suh. 2014. Higher Education, Gender & Work Dataset. `"https://www.pewsocialtrends.org/category/datasets/?download=20041"`

Etsuji Suzuki, Tomohiro Shinozaki, and Eiji Yamamoto. 2020. Causal diagrams: pitfalls and tips. *Journal of epidemiology* (2020), JE20190192.

Etsuji Suzuki and Tyler J VanderWeele. 2018. Mechanisms and uncertainty in randomized controlled trials: A commentary on Deaton and Cartwright. *Social science & medicine (1982)* 210 (2018), 83–85.

Amanda Swearngin, Andrew J Ko, and James Fogarty. 2018. Scout: Mixed-Initiative Exploration of Design Variations through High-Level Design Constraints. In *The 31st Annual ACM Symposium on User Interface Software and Technology Adjunct Proceedings*. ACM, 134–136.

Emi Tanaka. 2021. Edibble: An R-package to construct designs using the grammar of experimental design. https://github.com/emitanaka/edibble

R Core Team et al. 2013. R: A language and environment for statistical computing. (2013).

R Core Team and contributors worldwide. 2020. Package 'stats' v4.1.0. *CRAN* (2020). "https://stat.ethz.ch/R-manual/R-devel/library/stats/html/00Index.html"

Johannes Textor, Juliane Hardt, and Sven Knüppel. 2011. DAGitty: a graphical tool for analyzing causal diagrams. *Epidemiology* 22, 5 (2011), 745.

Inc. The MathWorks. 2020a. Matlab. "https://www.mathworks.com/"

Inc. The MathWorks. 2020b. Statistics and Machine Learning Toolbox. (2020). "https://www.mathworks.com/help/stats/index.html"

C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown. 2013. Auto-WEKA: Combined Selection and Hyperparameter Optimization of Classification Algorithms. In *Proc. of KDD-2013*. 847–855.

Walter Vandaele. 1987. *Participation in illegitimate activities: Ehrlich revisited, 1960*. Vol. 8677. Inter-university Consortium for Political and Social Research.

Tyler J VanderWeele. 2019. Principles of confounder selection. *European journal of epidemiology* 34, 3 (2019), 211–219.

András Vargha and Harold D Delaney. 2000. A critique and improvement of the CL common language effect size statistics of McGraw and Wong. *Journal of Educational and Behavioral Statistics* 25, 2 (2000), 101–132.

Priscilla Velentgas, Nancy A Dreyer, Parivash Nourjah, Scott R Smith, Marion M Torchia, et al. 2013. Developing a protocol for observational comparative effectiveness research: a user's guide. (2013).

William N Venables and Brian D Ripley. 2013. *Modern applied statistics with S-PLUS*. Springer Science & Business Media.

Lea Verou, Tarfah Alrashed, and David Karger. 2018. Extending a Reactive Expression Language with Data Update Actions for End-User Application Authoring. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. 379–387.

Chat Wacharamanotham, Krishna Subramanian, Sarah Theres Volkel, and Jan Borchers. 2015. Statsplorer: Guiding novices in statistical analysis. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 2693–2702.

Hadley Wickham et al. 2014. Tidy data. *Journal of Statistical Software* 59, 10 (2014), 1–23.

Hadley Wickham, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, Alex Hayes, Lionel Henry, Jim Hester, et al. 2019. Welcome to the Tidyverse. *Journal of Open Source Software* 4, 43 (2019), 1686.

Wikipedia contributors. 2019a. JMP (statistical software) — Wikipedia, The Free Encyclopedia. `https://en.wikipedia.org/w/index.php?title=JMP_(statistical_software)&oldid=887217350`. [Online; accessed 5-April-2019].

Wikipedia contributors. 2019b. R (programming language) — Wikipedia, The Free Encyclopedia. `https://en.wikipedia.org/w/index.php?title=R_(programming_language)&oldid=890657071`. [Online; accessed 5-April-2019].

Wikipedia contributors. 2019c. SAS (software) — Wikipedia, The Free Encyclopedia. `https://en.wikipedia.org/w/index.php?title=SAS_(software)&oldid=890451452`. [Online; accessed 5-April-2019].

Wikipedia contributors. 2019d. SPSS — Wikipedia, The Free Encyclopedia. `https://en.wikipedia.org/w/index.php?title=SPSS&oldid=888470477`. [Online; accessed 5-April-2019].

Chris J Wild and Maxine Pfannkuch. 1999. Statistical thinking in empirical enquiry. *International statistical review* 67, 3 (1999), 223–248.

Kanit Wongsuphasawat, Yang Liu, and Jeffrey Heer. 2019. Goals, Process, and Challenges of Exploratory Data Analysis: An Interview Study. *arXiv preprint arXiv:1911.00568* (2019).

Chapter A

Appendix One

A.1 Appendix section 1

Table A.1: Table in the Appendix