Spreadsheet Comprehension: Guesswork, Giving Up and Going Back to the Author

Sruti Srinivasa Ragavan
Microsoft Research, Cambridge, United Kingdom
t-ssr@microsoft.com

Advait Sarkar
Microsoft Research and University of Cambridge, Cambridge, United Kingdom
advait@microsoft.com

Andrew D Gordon
Microsoft Research, Cambridge, United Kingdom and University of Edinburgh, Edinburgh, United Kingdom
adg@microsoft.com

ABSTRACT

Spreadsheet users routinely read, and misread, others' spreadsheets, but literature offers only a high-level understanding of users' comprehension behaviors. This limits our ability to support millions of users in spreadsheet comprehension activities. Therefore, we conducted a think-aloud study of 15 spreadsheet users who read others' spreadsheets as part of their work. With qualitative coding of participants' comprehension needs, strategies and difficulties at 20-second granularity, our study provides the most detailed understanding of spreadsheet comprehension to date.

Participants comprehending spreadsheets spent around 40% of their time seeking additional information needed to understand the spreadsheet. These information seeking episodes were tedious: around 50% of participants reported feeling overwhelmed. Moreover, participants often failed to obtain the necessary information and worked instead with guesses about the spreadsheet. Eventually, 12 out of 15 participants decided to go back to the spreadsheet's author for clarifications. Our findings have design implications for reading as well as writing spreadsheets.

CCS CONCEPTS

• **Human-centered computing** → Human computer interaction (HCI); Empirical studies in HCI; • **Software and its engineering** → Software creation and management; Software post-development issues; Maintaining software..

KEYWORDS

Spreadsheets, end-user programming

ACM Reference Format:
Sruti Srinivasa Ragavan, Advait Sarkar, and Andrew D Gordon. 2021. Spreadsheet Comprehension: Guesswork, Giving Up and Going Back to the Author. In *CHI Conference on Human Factors in Computing Systems (CHI '21), May 08–13, 2021, Yokohama, Japan.* ACM, New York, NY, USA, 21 pages. https://doi.org/10.1145/3411764.3445634

CHI '21, May 08–13, 2021, Yokohama, Japan

ACM ISBN 978-1-4503-8096-6/21/05…$15.00
https://doi.org/10.1145/3411764.3445634

1 INTRODUCTION

The study of spreadsheets, and in general end-user programming, has a long history in the HCI community [11, 29, 64], with two Special Interest Groups (SIG): SIG End-User Programming and SIG End-User Software Engineering [5, 41, 57]. However, research and commercial efforts focus mostly on *spreadsheet authoring and editing*—making them easier and less error-prone. Disproportionately little attention is paid to perhaps an even more important aspect of spreadsheet use, namely *spreadsheet reading and comprehension* [32].

Spreadsheets are used by millions of people worldwide [56]. A survey of 1600 spreadsheet users found that only 12% of spreadsheet authors write spreadsheets exclusively for their own use; about 42% build spreadsheets that will be used by one or two other people, and another 45% write spreadsheets that will be used by several people [2, 58]. Moreover, 70% of the respondents reported sharing their spreadsheets with others. These results suggest that reading and comprehension is a common task among spreadsheet users, and we expect, based on the above evidence, that there are far more spreadsheet readers than there are spreadsheet authors.

Prior studies (e.g., [18, 22, 32]), mostly surveys and interviews, reveal that comprehending spreadsheets can be tedious and time-consuming; reasons include that large spreadsheets require a lot of scrolling, long and complex formulas can be hard to understand, and that spreadsheets often lack documentation [58].

Such difficulties in comprehension result not just in productivity loss for millions of spreadsheet readers, but also lead to errors, as studies of spreadsheet errors indicate [6, 12]. In fact, in a study of managers, spreadsheet miscomprehension appeared among the top 5 most frequently encountered errors: 25% of all participants reported miscomprehending spreadsheets, as did 50% of those working with complex models [6]. Miscomprehension during decision making can directly to lead to poor decisions, whereas miscomprehension during other activities (e.g., data entry, what-if analyses) could manifest as numerical errors, again resulting in inaccurate decisions or losses.

Both individuals as well as organizations recognize these difficulties of working with spreadsheets, as the advocacy and adoption for informal best-practice guidelines for spreadsheet design show [21, 58, 65]. A few large organizations—presumably, those that heavily rely on spreadsheets (e.g., accounting firms)—even formalize these guidelines [58], prescribing rules for formatting, layout, documentation, etc. that are then enforced on employees [40]. One such popular spreadsheet standard runs over 60 pages [13].

While guidelines and standards may provide a band-aid solution for large firms that can afford to invest in developing and enforcing

them their high expertise requirements and inflexibility make them an unviable option for most spreadsheet users. Some users also reported finding such guidelines and standards too stringent [58]. Moreover, we will see later in this paper that they only address a limited subset of comprehension issues, since spreadsheets have a lot that goes on "under-the-hood" (e.g., formulas, data validations). Therefore, we need better support for spreadsheet comprehension to be built into spreadsheet tools themselves.

To know what tool support to build, we must first understand what users currently do when they must comprehend a spreadsheet, and where exactly they encounter difficulties. Prior works (e.g., [18, 22]) offer some insights into spreadsheet comprehension difficulties, but they are mostly based on interviews and surveys with users, and these methods have limitations. They offer coarse-grained data which are useful at revealing broad issues, but they are not well-suited for revealing details such as the cognitive processes that happen inside a person's head. They also suffer from relying on participants' subjective memories of an experience, and a participant may not readily recall all details, especially interactions, tactics and annoyances that may have only lasted a few seconds. Such details, invaluable when building tools, can only be revealed by fine-grained data collection methods such as user observations or telemetry. Indeed, prior observation studies of spreadsheet users exist (e.g., [27]), but they were conducted in lab settings that do not represent real-world activities in terms of user needs, motivations, or prior knowledge of the domain. They also focus heavily on formula (or program) comprehension aspects.

Therefore, we conducted an observational study with 15 spreadsheet users thinking aloud as they comprehended and worked with unfamiliar spreadsheets as part of their real-world job tasks. We analyzed and coded these think-aloud sessions in 20 second segments, to answer the following questions:

- What information needs do users have during spreadsheet comprehension?
- What comprehension strategies do users adopt?
- What difficulties do they encounter along the way?

This paper makes the following contributions: 1) a novel treatment of spreadsheet comprehension as going beyond formula comprehension by spreadsheet developers, 2) the first study observing spreadsheet comprehension activities in real world tasks, confirming prior findings from the lab, 3) new insights into users' spreadsheet comprehension activities (e.g., that it involves information-seeking detours about 40% of the time, that the difficulties are despite efforts by the author to make the spreadsheet readable) , 4) new evidence that spreadsheet comprehension difficulties can lead to spreadsheet errors such as inaccurate formulas or incomplete data entry and 5) novel evidence that comprehension-related phenomena (e.g., confirmation bias, information seeking detours) observed in other domains, also occur in spreadsheets. These results have implications for spreadsheet tool building, as well as for theory building.

2 RELATED WORK

The release of Bricklin and Frankston's VisiCalc in 1979 is often considered to mark the beginning of the widespread adoption of electronic spreadsheets [66]. Just over 10 years later, Nardi and Miller found that, with spreadsheets, collaboration, such as between spreadsheet builders and end users, is the norm and not the exception [42], [43]—suggesting a dichotomy in the roles of spreadsheet authors and readers.

Subsequently, Hendry and Green interviewed spreadsheet users to ask: *"do you ever have trouble recalling how you structured a spreadsheet?"* [18]. They also asked participants to explain their own spreadsheet as they would to a colleague. The results revealed that spreadsheet users faced difficulties recollecting their own spreadsheets. Two further findings from this study have influenced most later work in spreadsheet comprehension: 1) users face difficulties understanding the formulas in the spreadsheet and 2) spreadsheets often lack critical background context needed to understand them; the context remains only in the author's head.

2.1 Understanding Spreadsheet Formulas

Understanding a spreadsheet's formulas can be hard for many reasons: 1) it is hard to see where cells draw their inputs from (even harder if they are from another sheet), 2) it is hard to gain a global view of the spreadsheet and 3) it can be hard to map formulas to their meanings.

One approach to address these problems is to use graphical representations of formulas and their dependencies, to reduce the cognitive burden of understanding them. Kankuzi and Sajaniemi built a visualization that provides a bird's eye view of the spreadsheet by clustering related cells [30]. Igarashi et al. built "fluid visualizations", a set of on-grid, static and animated visualizations, showing data flow across cells, tables and even the entire sheet [26]. In contrast, Shiozawa built interactive 3D visualizations where each cell reading input from another cell is raised to one level higher than its input cells: thus, a user can see the entire chain of computation as a three-dimensional tree [59]. Hodnigg and Mittermerier also built 3D visualizations for spreadsheet computations in three layers: one layer shows the formulas, the second layer shows the layout of the formulas on the grid and the third shows the flow of data among them [17]. Finally, Ballinger et al. built a set of visualizations addressing various comprehension needs of users: these include visualizations of data flow, sphere of influence of a cell and the depth of the calculation chain [1]. In addition to these research prototypes, most commercial spreadsheet packages provide visualizations such as "show precedents/dependents" to aid comprehension.

A well-studied spreadsheet comprehension issue is in cross-sheet cell referencing: since a user can only see one sheet at a time, they lack visibility into the data flow when a cell references another cell in a different sheet. To help with this problem, Hermans et al. built data flow diagrams [22] where different parts of a spreadsheet are represented as boxes, and a directed arrow between the boxes indicate the direction of data flow. A person can collapse or expand an arrow to inspect the data flow at different levels of granularity. Thus, Hermans et al. were able to satisfy four different information needs about data flow that they identified in "transfer scenarios", i.e., when a colleague transferred a spreadsheet to a person who must now maintain it.

Another approach to formula comprehension is to offer alternative notations—the idea being that it might be hard to see what cell references (e.g., C2) refer to. Examples of such notations include

natural language cell referencing [28], or mathematical notations as an alternative for formula syntaxes [33]. Sarkar et al. have also considered alternative notations, but for viewing and comprehending drag-filled formulas at once [54].

While the above works are about reader-end understanding, Hermans et al. consider that the problem can also be at the authoring end—that long and complex formulas can be error-prone and hard to understand and maintain. Drawing from the software engineering literature, they built a tool that uses simple heuristics to highlight long, complex and error-prone formulas (or "formula smells") on the grid [23]. The author could then address such formulas at their discretion, or with automated assistance [24].

2.1.1 Spreadsheet Comprehension As Program Comprehension. A notable body of spreadsheet literature treats spreadsheets as programs—relying on the observations that the spreadsheet formula language is a functional programming language, that spreadsheet formulas form a program, that spreadsheet users are end-user programmers, and that spreadsheet comprehension is comparable to program comprehension [5, 53]. Therefore, we briefly summarize the program comprehension literature here.

Early program comprehension research investigated the comprehensibility of various programming constructs [60] and notations [14, 44]. Researchers then began exploring the cognitive process involved in program comprehension. Key code cognition models are top-down vs. bottom-up models, opportunistic vs. systematic comprehension and the role of beacons and concepts in program comprehension; Storey [61] and von Mayrhauser [63] summarize these models. [63] also proposes a meta-model integrating the above key code cognition models.

These program comprehension theories, together with empirical reports from the field (e.g., [38]), have revealed several tool requirements to aid software comprehension. Much research focuses on turning these requirements into tool features. Storey's literature review categorizes such tools into three: 1) information extraction tools (e.g., extract information from various sources), 2) analysis techniques (e.g., static and dynamic analyses) and 3) presentation tools (e.g., IDE, visualization) [61].

Over the last decade, researchers have begun considering code comprehension as a fact-finding activity [35] and Lawrence et al. show the applicability of information foraging theory to how programmers navigate and comprehend code during maintenance activities [37]. Several researchers (e.g., [47, 48]) have since adopted the theory to building software comprehension and maintenance tools.

Since spreadsheet formulas are essentially programs, many of the results from program comprehension also apply to comprehending spreadsheet formulas. Just as programmers must comprehend bits and pieces of a program to accomplish programming tasks (e.g., debugging, maintenance), a spreadsheet user must also comprehend parts of the spreadsheet during spreadsheet programming activities (e.g., debugging, testing). To this extent, independent works of Burnett and Erwig on spreadsheet debugging and testing offer deep insights into spreadsheet program comprehension, or spreadsheet formula comprehension. Some key works are their lab studies exploring users' information needs [27] during spreadsheet debugging, sensemaking processes during spreadsheet debugging [15] and their explorations of how tools should communicate about spreadsheet faults to users, to aid debugging [36]. Appendix A shows the codes (especially from [27]) that overlap with our code set.

However, spreadsheets are not just programming tools, and spreadsheet users are not just end-user programmers building and debugging formulas. The versatile spreadsheet also serves as interface for data collection, data entry, decision making and data presentation, and often a spreadsheet user is also a normal end-user consuming the spreadsheet's results and data [12]. It stands to reason that the comprehension needs of a spreadsheet's end user (e.g., during decision making) will be very different from those of a spreadsheet developer (e.g., during debugging). In particular, the former might not involve any formula understanding at all. To this end, our treatment of spreadsheet comprehension is much broader than most prior works.

2.2 Supporting Contextual Understanding

Going beyond formulas, a finding of Hendry and Green's study [18] is the problem of missing context. When spreadsheet authors were asked to describe their spreadsheets as they would to a colleague, they talked about various bits of context needed to understand the spreadsheet (e.g., where the data came from and where it was going). But this knowledge resided only in participants' heads and was not captured in the spreadsheet, leading to potential difficulties for someone understanding the spreadsheet from scratch. In fact, participants sometimes struggled to recall details about *their own* spreadsheets.

Towards address this missing context problem, Hendry and Green hypothesized that a little documentation could go a long way. Therefore, as part of their CogMap system, they included lightweight ways of documenting regions of the spreadsheet grid. These little bits of documentation then show up on the grid, in context [19].

In contrast to such a lightweight approach, Canteiro and Cunha make the argument that one-size-fits-all documentation might not work for all spreadsheets: the needs for an end-user of a spreadsheet wanting to input data can be very different from the needs of a spreadsheet developer wanting to edit a formula. Their SpreadsheetDoc system, therefore, allows spreadsheet authors to add general documentation for the end user of the spreadsheet, as well as implementation details for use by the spreadsheet developer or maintainer [8].

Finally, in the last 5 years, Kohlhase et al. have conducted empirical inquiries into *what* counts as spreadsheet context [32]: when spreadsheet users were asked to explain their own spreadsheets, they used seven distinct kinds of knowledge items (e.g., definition, purpose, data provenance, examples) as part of their descriptions of a spreadsheet's cell or word or region. They also found that the original author of a spreadsheet offered much richer explanations than the users who had taken over spreadsheets from their original author—even though the latter may have used the spreadsheet regularly for several months. Following such evidence towards the need for capturing spreadsheet context, they built the SACHS system, which captures the semantics of the spreadsheet and uses the information to enhance the grid, to improve comprehension [34].

2.3 General Spreadsheet Readability

In addition to the above two aspects of spreadsheet understanding, namely formula and contextual understanding, a third aspect to spreadsheet comprehension is general readability. This includes the visual and logical layout of spreadsheets, styles and formats, colors, unambiguous row and column labels, etc. Although users' layout and formatting practices are largely ad hoc, some researchers have proposed "best practices" and guidelines [51, 65]. Other professional and standard bodies also offer elaborate guidelines for designing spreadsheets that are easy to understand [13]. But as we discussed earlier, they are neither applicable across the board [58], nor are they a solution for comprehending the various kinds of information in spreadsheets (e.g., data validations, conditional formatting rules, charts).

Several gaps in the existing literature should now be apparent. First, there are no prior observational studies of spreadsheet comprehension. Prior works (e.g., [18]) either offer less granular data from interviews and surveys, or they are observational studies that focus only on one aspect of spreadsheet comprehension, namely formula comprehension. Second, prior observation studies (e.g., [27]) were conducted in lab settings where participants worked on unfamiliar spreadsheets. While such studies offer valuable insights, the motivations and limited familiarity of participants with the spreadsheet's domain and context are not representative of the real world. Since such factors (e.g., prior knowledge, motivations) affect comprehension behaviours, those studies have limited validity. Other interview studies and surveys of real-world users also exist (e.g., [10, 55]), but do not deal with spreadsheet comprehension. Third, even though we know about the occurrence of spreadsheet comprehension errors from theoretical spreadsheet error taxonomies and empirical studies, we do not know why they happen or what they look like. Fine-grained data such as from user studies are needed to gather these details. Fourth, and finally, much of what we know about spreadsheet comprehension is largely based on the Hendry and Green study [18] that is now over 30 years old; the average end-user's experience of authoring and comprehending spreadsheets has changed substantially since then. Not only have the tools changed, but the applications of spreadsheets and the end-user population have massively broadened as well, and we need to revisit our assumptions. Only by studying contemporary tools and users, and by gathering complementary data such as through observation of actual comprehension activities, can we address this limitation. That is exactly what our study does.

Ours is the first observational study of users comprehending spreadsheets as part of real-world tasks. We analyzed participants' video and think-aloud data by qualitatively coding task videos at 20-second intervals. This offered detailed insights into participants' information needs when comprehending spreadsheets, what comprehension strategies they adopt, what difficulties they face, and why comprehension errors might occur.

3 METHODOLOGY

Two main considerations steered our study design. First, we needed fine-grained data from when users were comprehending spreadsheets, rather than post hoc. An observational user study was an easy choice for this. We adopted a think-aloud protocol to gain insight into participants' moment-by-moment comprehension processes.

Our second consideration was to choose a study task and spreadsheet with high ecological validity. This is tricky because spreadsheet users are experts in their domain and providing a spreadsheet from an unfamiliar domain, as in prior lab studies, will not work. Even if we provided them with a spreadsheet from their domain of expertise, it does not capture the familiarity with a spreadsheet's purpose, usage, task, organization and/or author that a user will have in real-world situations. Moreover, a fundamental limitation of lab studies is that they do not capture the real-world motivations of users. Therefore, we needed to observe people working on their actual work tasks.

However, finding participants willing to share their work spreadsheets can be hard, given that spreadsheets in organizations are important intellectual assets. Moreover, it is quite disruptive for participants to put their work on hold until a time suitable for researchers to observe them. These issues made it challenging to gain access to such comprehension episodes in the real world.

We were able to address the timing issue by being flexible and patient with recruitment: once participants were screened in, we worked with them to schedule a session where they were intending to do their task, thus minimizing disruptions to their work schedule. We were ultimately able to recruit suitable participants working on a range of tasks, over a wide range of domains (Table 1). Their spreadsheets had a wide range of complexity and size, from simple spreadsheets with only text and formatting, to complex ones with long formulas, spanning multiple sheets. Participants also had varied levels of experience and expertise in spreadsheets.

3.1 Recruitment and Participant Screening

We recruited participants in two ways: 1) we sent out recruitment emails to coworkers, family and other professional contacts (convenience sampling), and 2) we recruited participants via UserTesting.com, an online platform for conducting user studies. We screened participants on several criteria either using the screening feature on UserTesting.com, or over a 15-minute screening call. Eventually, we recruited 5 participants via email and 10 via UserTesting.com. All participants fulfilled the following criteria:

- Ecological validity: Participants had received a spreadsheet from another person that they needed to work on. Except P07, all participants had received their spreadsheet from a colleague; P07 received it from his partner.
- Minimal prior comprehension: Participants had not seen and understood the spreadsheet, or a similar one, before. All participants had some knowledge about what the spreadsheet was for, because they had to do something with it. We did allow participants if they had briefly glanced at their spreadsheets (without beginning the comprehension process) prior to the study.
- Need for comprehension: Participants believed that they would have to perform comprehension before they could use the spreadsheet (e.g., the use was not simply data input).
- Data sensitivity : Participants could bring the spreadsheet to the study exactly as received in the work context or could conceal sensitive data and still work on the task. All but one

participant (P12) brought their spreadsheets to the study as-is; P12 brought a spreadsheet with pseudo-anonymized data.
- English knowledge: Participants were required to have a working knowledge of English.

3.2 Study Protocol

The study sessions were run remotely, moderated by the first author. At the beginning of a session, participants were briefed about the purpose of the study and were asked to sign an informed consent form. Then, the participant answered a demographic questionnaire and introduced their task: what the spreadsheet was about, who sent the spreadsheet, and what outcomes they needed to achieve.

Participants were then introduced to the think-aloud protocol and were asked to work on their task for about 30 minutes, as they normally would, with the only exception that they think out aloud. Participants were made aware that they didn't have to complete the task within that time. We recorded the audio and the screen of the participants as they worked on their spreadsheet tasks.

After 30 minutes, participants answered the NASA TLX questionnaire [16], that we adapted to suit a comprehension task. The study then ended with a short retrospective interview, where participants were asked about: 1) how well they understood the spreadsheet, 2) which aspects of the spreadsheet were easy or hard to understand, 3) what aided or hurt their ability to understand the spreadsheet and 4) what they wished the spreadsheet had, that would have made their comprehension easier. We also used this interview to ask questions about the task follow-up (e.g., "you were stuck on X, how will you address that"). We recorded the audio and the screen for the interviews, and gathered the spreadsheet wherever we could. Each participant was compensated with USD 60 or equivalent in the form of payments (UserTesting.com) or electronic vouchers (other participants).

3.3 Qualitative Analysis

For this paper, we qualitatively analyzed the data from the participants' task videos. We coded screen actions as well as think-aloud verbalizations in 20-second segments. Since there is no prior code set specifically for spreadsheet comprehension (especially outside formula comprehension), we built our codebook from the data using open coding techniques [62]. However, our treatment of spreadsheet comprehension overlaps with prior work on spreadsheet debugging or spreadsheet context; as a result, wherever possible, we retained the code names and descriptions from prior work. The codebook in Appendix A lists the correspondences between our codes and those in prior work, if any. Some of our code definitions disambiguate and refine prior codes; for example, we split the code "provenance", from prior work, into: within the workbook (where is x?) and from outside the workbook (source). Our codebook also captures phenomena not captured in prior codebooks.

3.3.1 Building an Initial Codebook. We built an initial codebook as follows. We segmented each user study video into 20 second segments. We then picked 5 participants' videos (1/3 of the data) at random. For each participant, the first author (coder 1) and the second author (coder 2) independently coded: 1) comprehension and information-seeking activities, 2) comprehension strategies and tactics and 3) comprehension difficulties—allowing multiple codes per segment.

After each participant, the two coders discussed and revised the codebook by adding, removing, merging, splitting, and refining codes as needed. They then used the incrementally revised codebook to code the next participant. After completing this process for all five participants, the code definitions were relatively stable, with diminishing number of codebook changes (mostly, additions) per participant.

This stage resulted in a codebook comprised of 49 codes (17 activities, 19 strategies, 13 barriers). We reached a 70% agreement—consistent with prior studies with comparable code sizes [7]. We used Jaccard index as the measure of reliability since our codes were neither mutually exclusive, nor evenly distributed.

3.3.2 Coding Remaining Data. Our codebook was too large to achieve a satisfactory level of agreement. We had 49 codes across three categories, and we were coding both screen actions as well as think-aloud verbalizations. As a result, some coded segments had as many as 10 code assignments, and some codes occurred quite sparsely. As a result, during the initial coding, we found that coders often missed codes, leading to many of the initial disagreements. Campbell et al. report multiple instances of researchers encountering this problem in social sciences, suggesting that it is a common challenge when working with rich qualitative data. They also rightly note that combining codes to make the codebook manageable could lead to loss of information and offer various solutions instead to deal with reliability issues when coding using large codebooks. Following their recommendations, we chose the *negotiated agreement* approach to deal with coding disagreements that were largely coders' slips. In this approach, the data is coded by independent coders, followed by a negotiation process where all coders are present, to reach consensus on each codable event.

Next, coder 1 (who conducted the studies), coded all 15 participants' videos using the initial codebook. In this step, there was no need to modify or delete existing codes, but she had to add 22 new codes (8 activities, 8 strategies, 6 barriers) to capture new phenomena as they surfaced. The codebook eventually settled after 12 participants, consistent with prior studies in open coding [7]. Thus, the final codebook had 71 codes (25 activities, 27 strategies, 19 barriers).

After coder 1 had coded all videos, coder 2 reviewed each one of the coding assignments—keeping track of disagreements with existing codes, as well as suggestions for new code assignments that coder 1 had missed. At the end of the review, the Jaccard index agreement on the initial code assignments was 90.54% and there were no codebook changes. As expected, most of the disagreements came from missed code assignments. (Out of 4077 coder 1 assignments, agreed=4048, disagreed=29. Coder 2 made 462 new suggestions).

To ensure that the agreement was not solely based on chance, we computed Krippendorff's alpha coefficient, for a measure of inter-rater reliability. Since our data involved multiple codes per segment, we computed Krippendorff alpha to measure the agreement between the two coders in assignment (or non-assignment) of each individual code to a segment. The average Krippendorff alpha reliability across all codes was 0.92 (median=0.96).

Table 1: Participant and Task Demographics.

	PID	Gender	Age (yrs)	Functional area of job	Task	Spread-sheet
Email	P01	Woman	26-40	Compliance	A new colleague had prototyped a new process in a spreadsheet, and the participant wanted to understand it and use it to create an electronic Kanban board for a project.	Excel
	P02	Man	26-40	Data analyst	The participant was a new employee in an organization and wanted to enter his timesheets in the organization's spreadsheet-based timesheet system. He had seen the spreadsheet a month before when he received a walkthrough of the billing process in the company.	Sheets
	P03	Man	26-40	CS	The participant was a manager who received a summary of cost savings from his team; he wanted to understand the calculations so he could communicate them better to his upper management.	Excel
	P04	Man	26-40	Sci. / Engg. (non-CS/IT)	The participant had received a spreadsheet with data from scientific experiments conducted by a collaborator and he needed to look at the results to analyze them for a paper.	Sheets
	P05	Woman	41-60	Accounts/ finance	The participant worked for a real-estate company and wanted to understand the calculations for the investment return (IRR) projections made by a coworker.	Excel
User Testing .com	P06	Man	18-25	CS/IT	The participant received a spreadsheet from a client, who asked him to look at the personal exercise-training journal spreadsheet to see if he can make improvements to it.	Sheets
	P07	Man	26-40	Teaching planning	The participant received a family finance spreadsheet from his partner, who had asked him to enter in his monthly income and expenses.	Excel
	P08	Man	41-60	Project mgmt.	The participant's colleague had built a spreadsheet based on discussions with the latter, and the participant wanted to review and see if he agreed and to modify as he saw fit.	Excel
	P09	Man	26-40	Accounts/ finance	The participants' colleague had built a spreadsheet and the participant had to check the spreadsheet, since he had more experience; he also needed to enter in data that was not accessible to his colleague.	Excel
	P10	Man	18-25	Sales/Mktg	The participant worked on marketing strategy, and he had to create an Instagram marketing campaign for a client based on the spreadsheet his manager had built for an earlier campaign.	Excel
	P11	Man	26-40	Real estate	The participant worked in real-estate planning and their team wanted to create a spreadsheet for his managers to get a summary of the sites they managed. A colleague "had taken a stab" at the spreadsheet and the participant wanted to review it and offer suggestions.	Excel
	P12	Man	41-60	CS/IT	The participant received a spreadsheet with some data. He needed to clean and import into a database that he maintained.	Excel
	P13	Man	26-40	Accounts / finance	The participant needed to project the growth rate for a company based on their annual profit statements.	Excel
	P14	Man	18-25	CS/IT	The participant had to perform aggregate analysis on a public dataset (set of csv files) that he received from a colleague.	Excel
	P15	Woman	18-25	Sci./Engg. (non-CS / IT)	The participant's colleague ran some scientific experiments but suspected that something might be going wrong. So, she asked the participant to help her find it, since the latter had done similar experiments in the past.	Excel

The two coders then met to discuss and negotiate every coding disagreement and suggestion. The following code assignment changes were made because of negotiations. Coder 1 yielded to coder 2 on 361 assignments (29 deletions + 332 additions); coder 2 yielded to coder 1 on 121 assignments; there were 9 disagreements. Thus, we ended with 4380 code instances (4077+332-29). The final inter-rater agreement was 99.75% (Jaccard-index).

3.4 Limitations

Every study has its strengths and weaknesses. The greatest strength of our study is that our treatment of spreadsheet comprehension is much broader than prior treatments and that we observed actual spreadsheet users comprehending spreadsheets as part of their actual work or personal tasks. Moreover, our participants, their job functions, their spreadsheet expertise, and the complexity of their spreadsheets were diverse.

However, our study had only 15 participants, all English-speaking and working on only one task each, for limited time. This can be offset by triangulating findings with prior interview or survey studies that aggregate experiences over longer time or larger population or both, and conducting further studies to generalize our findings.

A second limitation is our qualitative analysis. We wanted to conduct in-depth analyses, looking closely at what participants were doing, and so we coded the videos in 20-second intervals. But the choice of 20 seconds is rather arbitrary: for phenomena spanning multiple segments, this worked well, but for other phenomena (e.g., look up something while filling in a cell) that lasted only a few seconds, this affected the fidelity at which we could capture phenomena. We still captured phenomena that lasted more than 5 seconds in a segment, but our discussions of time should be treated as approximate, as is the case with quantifying all qualitative data.

Third, as with any qualitative study, our data analysis required subjective interpretation of participants' actions—a process prone to biases. In our case, the large size of the codebook and the sparseness of the codes added an additional challenge to the reliability of our coding process. We attempted to minimize these limitations by building a codebook based on independent coding by two researchers, as well as by thorough reviewing and negotiations. We also make our codebook available for review and replication by other researchers (Appendix A).

Finally, as a word of caution interpreting numeric results in this paper—we emphasize that our study involved only 15 participants, working on *heterogenous* tasks: thus, the data is not sufficient for statistical inferences over a large population. Instead, quantitative measures such as code co-occurrences, or aggregates of time spent on various tasks, are only meant to guide hypothesis framing for future studies.

4 RESULT 1: SPREADSHEET COMPREHENSION INVOLVES OVER-THE-HOOD AND UNDER-THE-HOOD COMPREHENSION.

Participants performed all aspects of their work task during the study, and naturally did not limit themselves to just comprehension activities. Therefore, as Figure 1 shows, we coded both comprehension as well as non-comprehension activities (e.g., edits). Among the comprehension activities, participants spent about 60% of their time reading and understanding the spreadsheet's contents and interpreting it in their task context, *directly*—without having to seek additional information. As we describe in the rest of this section, this involved both over-the-hood and under-the-hood comprehension.

4.1 "Over-the-Hood" Comprehension: The Problems Of Hidden Data And Too Much Data

Over-the-hood comprehension is when participants engaged in understanding and interpreting what was visible on the spreadsheet: this includes text, numbers, computed values, charts and notes. (In Excel, a note is indicated by a red triangle in a cell corner; hovering the cursor over the triangle displays the text of the note). In our study, over-the-hood comprehension (codes: `understand + chart`) accounted for over 50% of participants' comprehension activities, on an average: majority of this time (42%) was spent reading the content on the grid (code: `understand`), and 10% was reading charts (code: `charts`). Individual participants spent as much as 80% of their time reading grid text, and 15% on charts (Figure 1 bottom).

Most of participants' over-the-hood understanding was aided by reading labels. Reading was largely systematic, as Figure 2 shows. Figure 2 is the code co-occurrence graph. Each node represents a code, the size and the darkness of the node represents how frequently the code occurred, across all participants (bigger node = greater code occurrence). An edge between two nodes indicate that the two codes co-occurred in the same segment: the thickness and darkness of an edge indicates how frequently the two codes co-occurred (thicker and darker edge = the two codes co-occurred frequently). The figure omits the least frequent co-occurrences (those below the 97^{th} percentile) for readability. The raw code assignment data is available in the auxiliary material for further analysis.

Observe, in the figure, the codes `understand` and `label`: the thick, dark edge indicates that the codes occurred frequently in the same segment, suggesting that participants' general grid-content reading was aided by labels. Similarly, notice the prominent edge between `understand` and `systematic reading` (denoted by `sys. read`) suggesting that the reading was largely systematic (i.e., a group of adjacent cells was read in sequence, rather than erratically/opportunistically reading cells located all over the grid).

Participants faced two difficulties in over-the-hood comprehension: the *hidden data problem* and the *too much data problem*.

4.1.1 The Hidden Data Problem. Since the horizontal and vertical extent of the grid in a spreadsheet is effectively unbounded (i.e., anything can be anywhere), there can be content outside of the visible area, perhaps even very distant from the rest of the grid—presumably done so by the author to avoid clutter. This often induced an information need, namely, to know the *extent* of the grid that is being used. As Figure 1 (top) shows, 9 out of 15 participants engaged in this activity (code: `extent`) to ensure that there were no hidden parts that they had missed, but this was an informal, manual, and non-exhaustive process. It is known that *formula* dependencies suffer from low 'visibility' [18] (per the cognitive dimensions

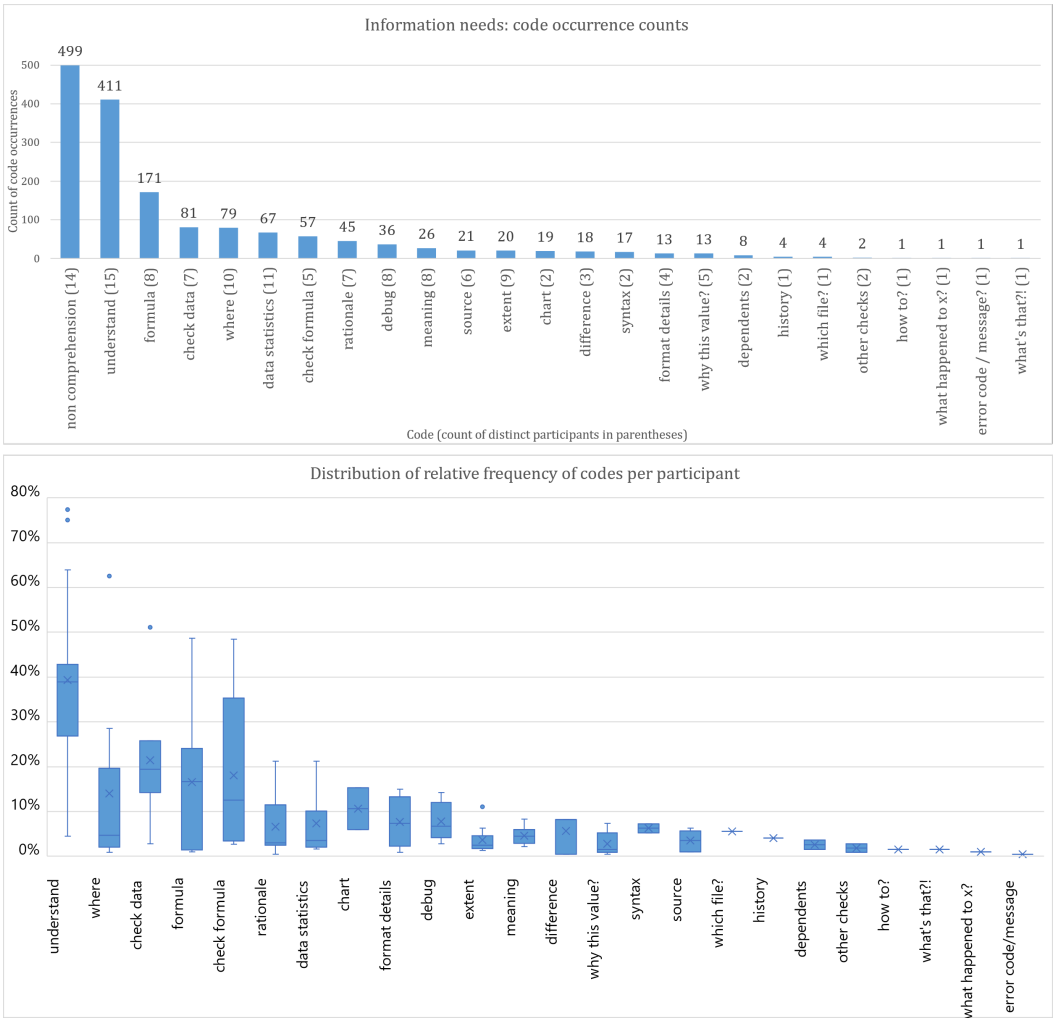

Figure 1: *Information needs.* Top: Code occurrences (x-axis=information seeking code, with no. of participants the code occurred in. Y-axis = total no. of times the code occurred). Bottom: Distribution of relative frequency of a code across participants (excluding the non-comprehension code for clarity). Participants spent a high fraction of time (as high as 80%) reading the grid contents directly—without any information seeking tangent; all 15 participants engaged in this fundamental activity in spreadsheet comprehension (code: understand).

framework [3]); our new observation is that the spreadsheet grid itself suffers from the visibility problem.

> P05: "I don't know where that's coming from . . . [scrolls all the way to the bottom] . . . it doesn't seem to be coming from the other sheet, it must be coming from this sheet, looking around [scrolls all the way to the right] . . . I can't see anything that's helpful, any hidden cells or anything [and scrolling all the way again] . . . I don't know where that's from, that has flummoxed me. . . must be kind of somewhere."

4.1.2 The Too Much Data Problem. Sometimes, participants had to read and interpret the *data* in the spreadsheets, rather than just labels and summaries. However, a spreadsheet sometimes contained so much data, overwhelming participants (P07, P10, P13). To deal with this, participants often looked at various kinds of statistics for the data, either by computing them using formulas (P04), or by manual inspection (P04, P15), or by using the status bar summary at the bottom (P04). One participant, P04, created charts for the data and two other participants (P11, P15) wanted to tell the spreadsheet's author to add charts. When asked "what do you think would have made the spreadsheet easier to understand?", P04 responded:

> ". . . [I would like an] option for me to find the most significant columns based on the weightage of the rows ... most columns have a lot of zeroes, so that could be pushed to the right or even out of the view . . . columns with more . . . continuous data... that is what matters in most of the cases."

4.1.3 "Under-the-hood" Comprehension Goes Beyond Formulas. As expected, participants' under-the-hood comprehension involved formula comprehension. As Figure 1 (bottom) shows, 8 participants

spent time reading and comprehending formulas *in situ*, without having to leave the context to seek additional information; this accounted for about 18% of their total task time (code: `formula`). For one participant, P05, who had a large workbook with long and complex formulas, `formula` accounted for about 50% of all codes. Notice the long tail ending at 50% for `formula` in Figure 1 (bottom) corresponding to P5, and the large node for `formula` in P05's code co-occurrence graph in Figure 3 (top).

In addition to simply reading formulas and references, participants also often spent time seeking additional information to understand formulas (e.g., to understand a function); thus, the actual formula comprehension times are much higher.

However, formulas were not the only under-the-hood contents participants had to understand. During data entry, P02 encountered a data validation error, and he spent time understanding and recovering from that error, when he did not know that a rule for that cell existed in the first place. Another 4 out of 15 participants (P06, P10, P12, P15) looked for how cells were formatted as part of their information seeking activities (code: `format details`). Here, participants wanted to figure out what colors and layouts applied to a cell, as well as conditional formatting rules. For example, P06 spent significant time trying to match the formatting of a new table he was authoring to an existing one in the spreadsheet, considering font size, colors, cell orientations, and borders.

> P06: ". . .I also want to understand how exactly these are being colored. I imagine it is conditional formatting ... I see that it is set to each color based on what the value in the cell itself is equal to. . . . is this also conditionally formatted? ... these just say that if it is not empty it will automatically be green. . .. [after some time] ... I don't know exactly how this text is floating."

4.2 Ensuring Correctness Requires Over-The-Hood And Under-The-Hood Inspections.

To ensure the correctness of spreadsheets, participants had to inspect under as well as over-the-hood. Sometimes, participants wanted to check the *accuracy* of only the data in the spreadsheet; six participants (P07, P08, P09, P10, P12, P13) engaged in some form of data checking activity, and for P09 and P12, the `check data` code accounted for 51% of all codes. Checking the accuracy of the data was mostly an over-the-hood activity.

In contrast, checking the accuracy of formulas (code: `check formula`) involved both over-the-hood inspection to ascertain whether the computed value was accurate, as well as under-the-hood inspection to ensure that the logic was accurate. Since formulas are often drag filled across multiple rows or columns, participants also checked whether the same formula was drag filled accurately. For example,

> P09: "I usually check these tabs (referring to cells) and I see in here (pointing to formula bar), if something changes then it has not been done correctly."

Once he had checked a set of formulas, P09 then cross-checked the formula output using the Excel status bar, which can be configured to display various summary statistics computed on the current grid selection:

> "I also like to highlight these and see if the sum in here (pointing to a cell with formula) matches the amount in here (pointing to status bar).

After verifying the formula output in this manner, P09 proceeded to check the formulas in the next part of the table—thus, alternating between checking what was over-the-hood and what was under-the-hood. A total of 11 participants engaged in some form of data or formula checking activities. Additionally, P01 had to check for whether the links to documents in the spreadsheet worked (code: `other checks`).

5 RESULT 2: SPREADSHEET COMPREHENSION REQUIRES INFORMATION DETOURS.

The activities of reading, understanding, and interpreting the contents of the spreadsheet—below and above the hood—accounted for only 60% of participants' spreadsheet comprehension activities. The remaining 40% of the time was spent in *information seeking*, where participants had to look for additional information needed to better understand or work with the spreadsheet's content. Figure 1 shows the codes pertaining to information needs, and the relative frequency of their occurrence across participants.

These information-seeking activities were sometimes triggered while comprehending parts of a spreadsheet—above- and under-the-hood. Other times, they were a result of non-comprehension activities, when participants needed additional information to make progress (e.g., "where should I enter this data?"). In the rest of this section, we describe the broad categories of participants' information needs.

5.1 Information Needs: Locating "Stuff" Within Spreadsheets.

Some information needs could be satisfied with what was already present in the spreadsheet. Participants simply had to locate them—but that required leaving the current task context, searching, and then resuming their previous activity. Such information needs included: 1) `where` is X? 2) `dependents` of a cell, 3) what `formatting details` apply to the cell and 4) are there hidden cells or tables? (`extent`). Answering these questions requires going both above and under-the-hood.

Of these needs, the question "where is X?" was the most frequent, encountered by 10 out of 15 participants. Figure 1 (top) shows `where` was among the top 5 information needs. Further, as the boxplot in Figure 1 (bottom) suggests, `where` accounted for as many as 60% of codes for one participant, namely P14.

> "I have seen those [multiple sheets] on Google sheets, but in this case, I was just overwhelmed with the information . . . I didn't even notice that there was something there".

Difficulties in answering "where" questions sometimes led to errors. For example, P14 wanted to write formulas in one sheet to aggregate the data in a second sheet. For this, P14 repeatedly asked the question "where (what rows) does data for this category

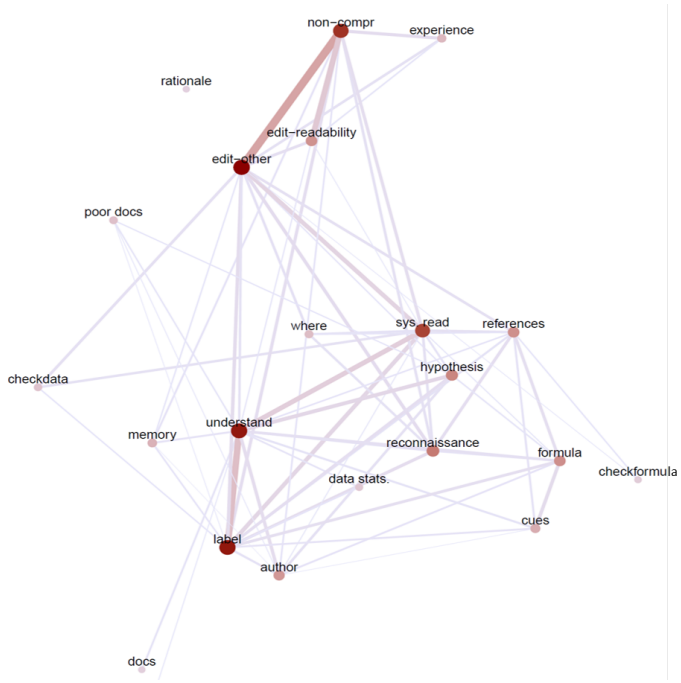

Figure 2: Code co-occurrence graph, showing top 2% frequent code cooccurrences. Node=code, thickness of edge = frequency of the cooccurrence of the two codes in the same segment. Consider the thick edges between nodes "understand", "sys. read" for systematic reading, and "label": when reading the grid's over-the-hood contents (code: understand), participants were largely systematic (code: sys. read) and relied on labels. For under-the-hood formula comprehension, consider codes formula, references, cues and reconnaissance and the edges between them: here, participants mostly read cell references and used cues to highlight where the references existed, or navigated to the cell to see what it meant (code: reconnaissance).

begin and end?", so that he could specify the range in formulas (e.g., A121:A168). But the tedium of navigating back and forth between sheets, scrolling to locate where a category began and ended, and memorizing and recalling the row numbers while switching sheets led to errors. On multiple instances, P14 either misread or misremembered row numbers, resulting in incorrect formulas (e.g., A168 instead of A167). Since P14 did not spot those errors, he ended up with inaccurate results.

Another instance of such an error was in P07, who needed to enter the expenses for each month in a personal finance spreadsheet received from his partner. P07 spent significant time unsuccessfully trying to locate where to enter the expenses, and eventually gave up. They were found in another sheet, and when prompted during the retrospective interviews, P07 reported not knowing that the multiple sheets feature also existed in Microsoft Excel, even though he had seen them in Google Sheets. Here again, difficulties answering "where" questions about a spreadsheet's contents resulted in incomplete data, a potential source of errors.

We argue that looking up parts of a spreadsheet for data entry or to enter formulas are in fact micro-comprehension activities embedded in other larger activities. Examples such as the above offer empirical evidence of miscomprehension errors leading to other kinds of spreadsheet errors. Building good comprehension tools, therefore, is important to guard users against misreading/miscomprehension errors, and in turn against other kinds of errors (e.g., formula errors, reuse errors, data entry errors).

5.2 Information Needs: Deducing How "Stuff" Works Together.

The previous category of information needs was about locating the pieces in the spreadsheet (e.g., where is X? where is X used?). In contrast, this category is about how pieces work together. In our study, these information needs included: 1) why is X this value, and not something else that I expect it to be? (code: `why this value`) 2) Is there a difference between X and Y, and if so, what and why? (code: `difference`) 3) why is this an error? (code: `debug`). These questions were asked in the context of data, formulas, as well as charts. They also arose as part of debugging—which some participants spent over 10% of their task time doing—as well as when simply trying to understand how something works. These results are consistent with prior findings on information seeking during debugging activities, both in spreadsheets and professional programming [15, 27, 37].

5.3 Information Needs: Tracing Formula "Rabbit Holes"

Understanding formulas entailed locating pieces of information as well as putting them together. Since formulas are typically written using cell addresses (instead of labels or names, which some spreadsheet packages support), participants had to first grasp what the address contained by navigating to the cell address and reading the text labels in adjacent cells. For example, to understand what is

being summed in the formula =SUM (A1:A100), one needs to visit the range A1:A100 and read the labels adjacent to it. This is analogous to comprehending a sentence "*I am going to meet the person in 1, CHI Way*": one needs to look up who lives in *1, CHI way* to infer that the sentence means "*I'm going to meet Potter*". Moreover, for a formula with many references, one needs to maintain all these labels in working memory to understand the formula in its entirety.

To a large extent, participants in our study used tool-provided cues to understand what cell references meant: by double clicking a formula, the tool highlighted referenced cells on the grid, which gave participants a clear target for visual search and enabled them to quickly locate relevant labels. In Figure 2, the prominent lines between the code `cues` and the codes `formula` and `check formula` indicate that participants often attended to these cues when understanding formulas.

However, these affordances were often of limited use, especially when the spreadsheet was large: either the highlighted cells were not visible (e.g., off-screen or in a different sheet), or the labels were not visible (e.g., the author had not provided a label, or there was only a single label for a large span of rows or columns and it was off-screen), or the highlighted cell also contained formulas that participants had to understand. Therefore, users needed to navigate away from the formula they were comprehending, to these cell addresses, to try and read labels and understand what the formula meant. The code `reconnaissance` captures such navigations, and as Figure 2 shows, `reconnaissance` frequently co-occurred with `formula` and `label` codes.

The formula comprehensions of P05 exemplified these difficulties in interpreting formulas and understanding cell references. P05 had to comprehend a large financial spreadsheet with several long and complex formulas (using NPV, XIRR, VLOOKUP and several levels of nested IF); the outlier in Figure 1 (bottom) for the code `formula` indicates the heavy formula reading by P05, as does Figure 3 (top), where `formula` is the most prominent in P05's codes. As she went about comprehending formulas, she needed to scroll and navigate to other places about as many times as she used cues to highlight cells. Notice the high co-occurrence of the code `reconnaissance` with codes `formula, references` and `label`, indicating frequently having to perform context switches to look up references or labels.

Several of these instances were due to what P05 referred to as the "*rabbit hole*" (P05) of formulas referencing other formulas—sometimes, she lost her way and had to find her way back. As a result, her formula comprehension was accompanied by sighs and remarks such as *"This is so annoying"* and *"Oh, my God!"*. The codes `formula` and `overwhelmed` co-occurred for two participants (P04, P11) in a total of six segments.

These results are at once unsurprising and surprising—unsurprising because Hendry and Green reported these findings in [18], and surprising because, even though [18] highlighted these problems way back in 1994, the problems persist in commercial spreadsheet tools and affect hundreds of millions of users.

5.4 Information Needs: Understanding Unfamiliar Spreadsheet Concepts.

A fourth category of comprehension needs was to understand unfamiliar concepts that participants encountered when performing their tasks. Often these meant understanding unfamiliar functions that participants encountered in formulas. But when they attempted to comprehend them, they often encountered difficulties. For example, P05 reported that function names were non-intuitive, and rightly so. She saw the function name SUMIFS and interpreted it as "sum I-F-S". Only on reading the function's description in the documentation did P05 realize that it is in fact "sum if(s)" (i.e., "IFS" was the plural of "IF" and not a three-letter acronym). In other instances, participants read the function names correctly, but didn't know what the function did. They read tooltips, documentation, in-Excel and even searched the Internet (often unsuccessfully) to seek answers. P05 eventually decided to go back to the author to gain a complete understanding of the formula:

> P05: "well, I understand what he's come up with, but I can't understand how. . . that formula was beyond me. So, I think I'll move on."

During the retrospective interview, she said:

> P05: "I need to check with the author about the XIRR one. . . for the other, I think I understood . . . the one with that INDIRECT thing, it was just trying to get the name, it doesn't really matter too much, but I'd like to be able to see, to check everything."

Unfamiliar concepts also went beyond formulas. P07 did not know why a cell contained "#####", assumed it was an error, and wasted time trying to debug it. In Microsoft Excel and Google Sheets, "#####" in a cell means that the cell is not wide enough to display a number in full—presumably to prevent miscomprehension. Ironically, P07 miscomprehended that cue and abandoned going down his current task path after unsuccessfully trying to understand this feature. Even with familiar functions, participants had difficulties when the function was used in an unfamiliar way; for example, P11:

> "VLOOKUP. . . he's done a *VLOOKUP* from somewhere. . . if I see VLOOKUPs I get a little wary, I'll come back to it later because I think something complicated is going on."

These results raise the question: how can we design tools that provide such information about the tool (essentially, help and documentation) in a manner that is: 1) relevant to the context of what the user is doing, 2) accommodates a range of users with varying backgrounds and levels of expertise, 3) does not disrupt the user's workflow and 4) is known to the user (i.e., the user shouldn't have to spend time learning about the learning tool). More broadly, the design opportunity is to convert information-seeking events such as these into teaching moments, providing *minimal*, *in-context* instruction to the user, thus helping them gain a mastery over spreadsheets, over time, one bit at a time [9].

5.5 Information Needs: Background Context That Lies In People's Heads.

Finally, some information needs were about the context in which the spreadsheet was created (e.g., definitions and assumptions, formula explanations) and the context in which it will be used (e.g., how to interpret a value). Prior studies [18, 32] have found that spreadsheet users, when explaining their own spreadsheets, provide both these

kinds of contextual information, but do not capture them in their spreadsheets.

In our study as well, spreadsheets did not capture a lot of the background context such as "how to interpret a value", or examples of what each item meant. However, since participants brought their own work tasks and the associated spreadsheets, they had an idea of the context in which the spreadsheet was created, how to use it (or how it will be used) and what they needed to do with it and why. They described such context at the beginning of the study, as well as part of the verbalizations during the study:

> P04: "CPI and IPC are reciprocals of each other."

Yet, participants often missed critical contextual information needed as part of their comprehension activities. These information needs included: 1) `rationale` for why something is done (e.g., why was this highlighted?), 2) `source` of data (i.e., where is the data that this number is calculated based upon?), 3) `meaning` (e.g., what is "E2E"? What is the significance of the hard-coded constant "53400"? What do the colors and the highlights mean?) and 4) `history` (e.g., what has changed from what I knew before?). Together, these context-seeking codes (`history` + `source` + `rationale` + `meaning`) occurred in 10 participants and accounted for an average of 10% of their comprehension activities; for P15, this was almost 30%.

Participants often tried to infer missing context based on what they knew and what was visible in the spreadsheet. For example, P15 inferred the units of measurement for a column as meters by simply glancing at the values; even adding it to the column header, presumably to facilitate subsequent comprehension.

> P15: "I did the experiment ... so maybe these are heights, these are the masses and then 'I' would be the inertia [edits the spreadsheet to make the labels clearer] . . . guessing this is in meters and this is in kilograms, just from looking at the values . . ." [adds (m) and (kg) in the spreadsheet].

Participants also frequently turned to `documentation` within the spreadsheet and the code `documentation` co-occurred with 45.7% of context-seeking codes. Such context documentation existed in most spreadsheets we encountered, in the form of comments, text in cells, labels (an implicit form of documentation), and formatting. For example, right after figuring out what the measures and units were, P15 said:

> "I did this experiment some time ago, so I'm just trying to recall, like, why there would be two values for height . . . [scrolls, finds some documentation] I had used ... ok wait a minute . . . ok, so here she is saying use reported mass and height . . . I guess these are the reported and the actual, I do need to ask her which one is which . . . or look at my own old files."

However, despite efforts on the part of the spreadsheet author to make the spreadsheet more comprehensible, participants faced difficulties gathering contextual information from the available documentation, formatting, and highlights: the code `poor documentation` frequently co-occurred with context-seeking codes (31%), across 6 out of 15 participants (P01, P04, P05, P07, P11, P15). In some cases, ad hoc formatting (presumably meant to aid comprehension) appeared without documentation.

> P11: "I don't know why he has highlighted this in yellow, it's the same formula, [scrolling] I don't know why he's highlighted this in yellow either [scrolling] and some more orange . . . this is red. I don't know what the highlights are. I'll have to reach out to him."

These instances of poor documentation do not occur because authors are unwilling to spend effort on documentation, layout and formatting. On the contrary, all the spreadsheets we witnessed contained elements that indicated effortful consideration of comprehensibility from the author. For example, in the case of P11, the spreadsheet was laid out across several sheets, formatted with colors and labels and had a summary page—the author had put in efforts to make the spreadsheet adhere to conventional best practices, perhaps at the expense of documenting contextual information. Similarly, we discovered that the author of P03's spreadsheet had documented a critical piece of information only in an email alongside—rather than within—the spreadsheet. Prior studies [58] show that about 20% of spreadsheet authors write documentation in places outside the spreadsheet, suggesting more fundamental disincentives or even barriers to documenting within the spreadsheet. We discuss some potential reasons for this in section 7.

6 RESULT 3: SPREADSHEET COMPREHENSION IS REPLETE WITH GUESSES AND GOING BACK TO THE AUTHOR.

We've seen that not only do spreadsheets frequently *not* contain the information participants need, but also that even if the information is present, many difficulties are encountered while seeking it. In many cases, the information seeking episode ended in failures. On various occasions, participants did not understand unfamiliar features and functions even after consulting online resources and in-application documentation, they did not ascertain the extent of the spreadsheet (e.g., the next sheet; P07), or there was just insufficient information in the spreadsheet (e.g., due to limited documentation; P14). Figure 4 summarizes the barriers, and the number of participants that encountered each of those information-seeking barriers.

When missing critical information prevented them from making progress on their task, participants resorted to one of two tactics: 1) forming hypotheses about the author and their context (instead of just about the spreadsheet), or 2) going back to the author to seek information.

Forming and testing hypotheses was a common coping strategy. Prior work has shown that forming and testing hypotheses is central to tasks such as spreadsheet debugging [15], and in general to human sensemaking in information-intensive tasks (e.g., program comprehension during maintenance [37], text comprehension for research [4]). Many participants' comprehension and information seeking were driven by hypotheses about their spreadsheets, based on what they had already seen, or knew about the domain.

> P11: ". . . he's equated that to a 100% seat capacity buffer, I don't know what that means. . . oh, 100% seat capacity or buffer. . . if I change it here [makes edit],

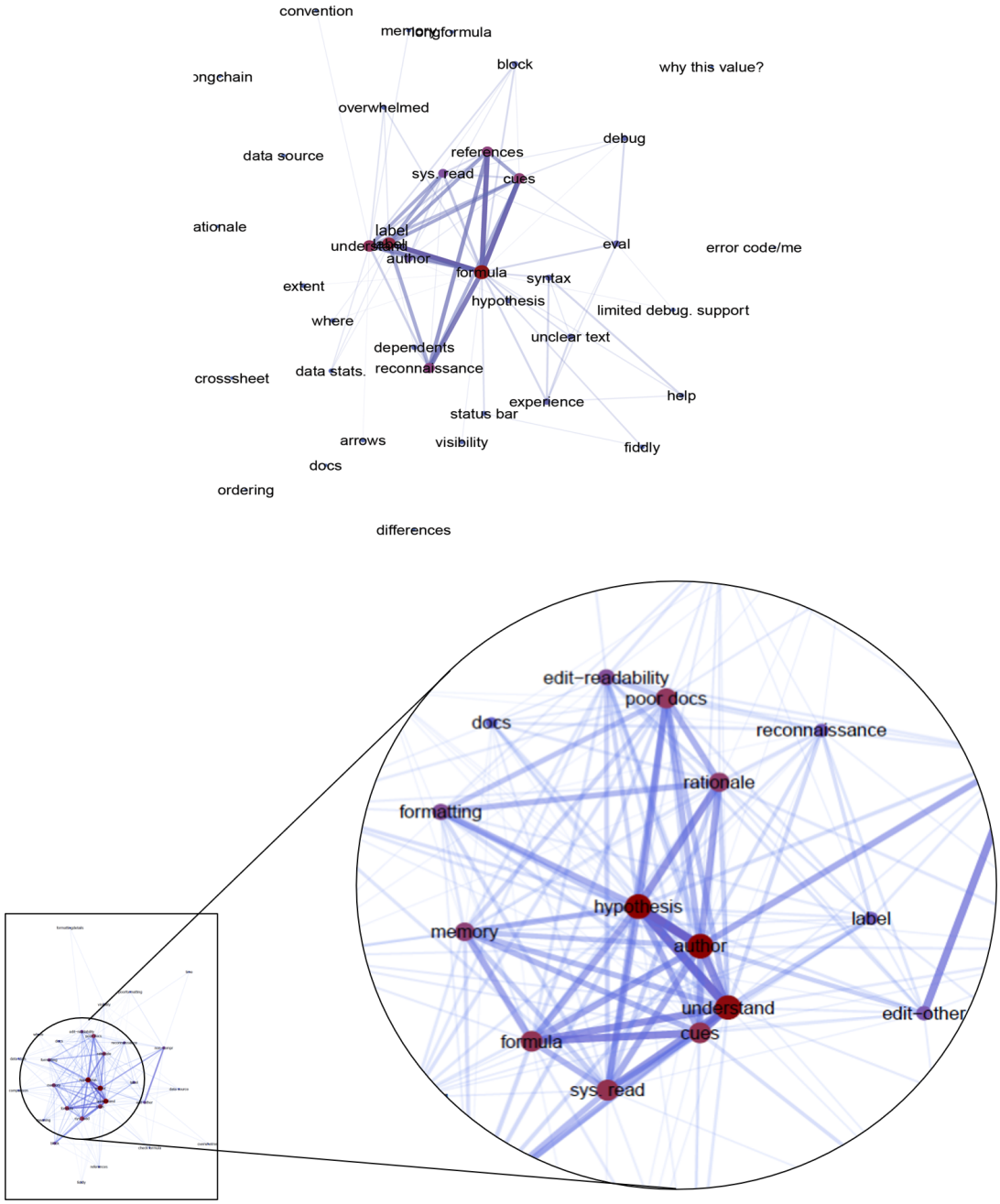

Figure 3: Code co-occurrences for P05, P15. Top: P05 was a heavy formula reader and engaged in frequent reconnaissance to look up labels for cell references in formulas. Bottom: P15's spreadsheet missed a lot of contextual information (e.g., rationale, formatting) and so she formed hypotheses about what it is that the author might trying to communicate through the spreadsheet. (docs=documentation).

> then it changes there [sees changes], so that's what he means."

However, the same barriers to information seeking that we have previously discussed also adversely affected participants' ability to gather the evidence needed to test their hypotheses, frequently leading them to simulate the spreadsheet's author—considering in what context the latter created the spreadsheet, for what purpose, guessing what the author might be trying to communicate given the task context, and the extent of the author's knowledge about the domain. One such excruciating example was P15 hypothesizing why the author had highlighted some values:

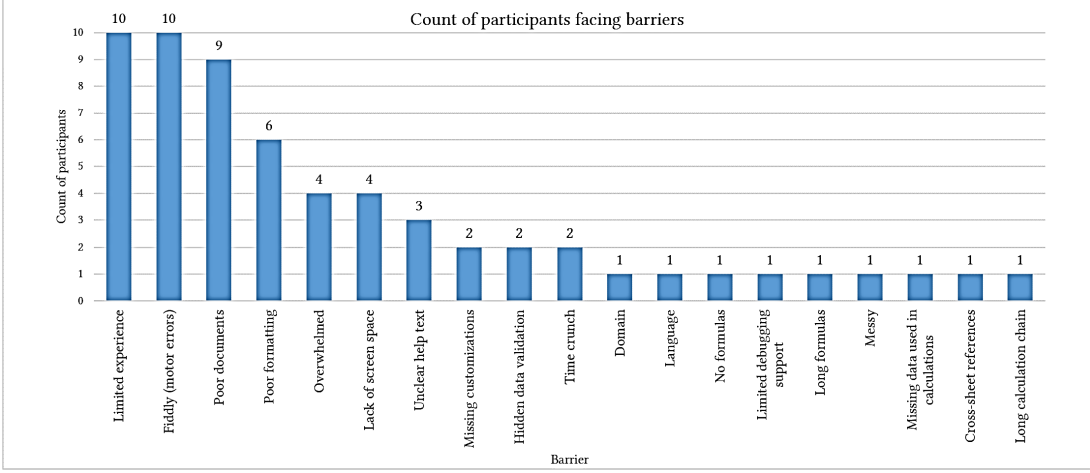

Figure 4: Barriers. Participants faced several barriers when comprehending spreadsheets.

> P15: ". . . output 1 if they're the same, or output 2 if they're not ... They're all the same except for this value. Maybe that's why it was highlighted in red . . . Again, this one seems to be different and that's probably why it's highlighted in red . . . I guess she had to round because there were very small differences. . . I guess she only wanted the large differences . . .These are small, but these are big, so I guess that is why she highlighted them in red. . . pretty large that's why it's also highlighted in red. . . I'm guessing these are highlighted in yellow because there are these reds . . . [scrolls around, sees a new column] I guess these are subject numbers, so I'll just write it there. . . [goes back to checking highlighted rows] but I don't know why this one is. . . [highlighted in yellow but does not contain a red cell]. I guess she didn't do the differences here, maybe she did that in her head . . . [checks values to test hypothesis] . . . This is a problem, but I guess this is already highlighted and so she left it like that. I think what happened is, for these parameters, she had come in and highlighted the cells in red . . . and then she went back and looked at these differences, I'm guessing just manually, and then picked out this row too because it was 200. [checks more rows to test hypothesis] . . . To me this is pretty large too [highlights in green and adds a legend] ... I guess she was just doing some high level and not the nitty gritty. . .."

We coded such verbalizations with both `hypothesis` (indicating the tentativeness in the participants' understanding) and `author` (indicating that the participant was considering the author, not just the spreadsheet). These two codes co-occurred frequently in participants such as P11 and P15. Code co-occurrences for P15 are shown in Figure 3 (bottom): note the prominent edge between `hypothesis` and `author`.

The problem with hypotheses about what the author did is that they can only be verified by the author. This includes confirming the tentative conclusions of participants, as well as the chain of evidence leading up to it, as the definition of hypothesis as "*tentative representation of conclusions with supporting arguments*" suggests in the sensemaking literature [49]. Appropriately, therefore, 10 out of 15 participants in our study wanted to go back to the author to seek additional information about the spreadsheet. However, research suggests that due to the limitations of working memory, people cannot keep track of too many hypotheses and their evidence (and thus might miss testing them, resulting in erroneous judgments). Sometimes, even if the conclusions were right, the reasoning behind the conclusions may be incorrect and lead to errors, including erroneous decisions due to confirmation bias [49].

Evidence from the retrospective interviews (conducted immediately after the task) suggests this might be a source of spreadsheet comprehension errors. Participants mentioned wanting to ask the author about missing information in the spreadsheet (that they had no way of even guessing), but not about confirming their hypotheses, or their reasoning. In the following example, P01 formed several hypotheses about the spreadsheet: that it might not be relevant to their organization, that the various tabs (sheets) mapped to "lifecycle stages", and that the author had done something because he had only joined the organization recently (and not due to another reason). All these assumptions could only be confirmed by the author.

> P01: ". . . not sure if we even have [these kinds of] agreements, . . . this is a new term, not sure if it really applies to us . . . it appears that the team wants to add some structure to their process . . . news article looks like he's using a few different things. . . he has added in examples of risk management, interesting, . . . for me, it looks like this spreadsheet is intended to manage the lifecycle of the projects. This person is new to the company, so many of the details might not be relevant, but I can see how some of them might be helpful."

She then proceeded to work on her task for several minutes, translating the spreadsheet details to a project Kanban board. Then,

at the end of the study, she said during the retrospective interview that she needed to confirm only one of these hypotheses that she had verbalized during the think aloud;

> P01: "the intent of the spreadsheet and whether the tabs are stages in the lifecycle as she understands."

Only three participants (P04, P11, P15) kept brief notes as they went along to record the things they wanted to check with the author. Moreover, all participants began making progress on tasks as they awaited an opportunity to confirm their understanding with the author, thus needing, for the sake of efficiency and practicality, to work with unvalidated assumptions.

7 DISCUSSION & IMPLICATIONS FOR TOOLS

Comprehension is central to all spreadsheet activity. Based on the intuition that there is more to spreadsheet comprehension than just formula comprehension (or program comprehension), we conducted the first observational study of how user comprehend spreadsheets in the real world.

Our results confirm that spreadsheet comprehension is indeed much broader than formula comprehension, and reveal several challenges to users' spreadsheet comprehension activities: 1) information is hidden away from plain sight in other parts of the grid, or under-the-hood (e.g., data validation rules), 2) users frequently have to switch context to seek additional information needed to comprehend spreadsheets, 3) the lack of critical information needed to comprehend a spreadsheet to make progress on a task led participants to form and test hypotheses about the missing pieces of information, 4) the frequent failure of even the hypothesis testing strategy due to lack of information needed to test the hypotheses and 5) the need for spreadsheet users to seek out the spreadsheet's authors to answer questions about the spreadsheet. Some of these results have been observed in prior spreadsheet studies in limited context as in program comprehension, or spreadsheet debugging literature (see Appendix A for phenomena that also occur in prior studies). Our study offers novel evidence that these also occur beyond just formula comprehension.

Our study also provides evidence that users face these difficulties *despite* the efforts of the author to make the spreadsheet readable, and that these comprehension and information seeking difficulties lead to miscomprehension errors which might manifest as other kinds of spreadsheet errors. These results have implications for spreadsheet tools, and more fundamental theories, for both spreadsheet authors and readers.

7.1 Implications: the Reader Side of the Equation

Spreadsheet users need to comprehend what is over-the-hood, as well as under. In over-the-hood comprehension, participants read the data and the charts on the grid and faced two kinds of problems. The *hidden data* problem arises when relevant portions of the grid are difficult to navigate to and bring onscreen, and the *too much data* problem arises when participants are overwhelmed by the amount of content in the spreadsheet and are unable to make sense of it easily. Thus, there are opportunities for summarizing the spreadsheet or drawing attention to hidden parts of it.

During under-the-hood comprehension, participants faced difficulties understanding formulas, keeping track of references, and having to navigate to cells to see their labels. Whereas even professional programmers have largely moved away from pointers and direct memory addresses (e.g., 0x123456) because of their error-proneness and comprehension difficulties [25], spreadsheet users (who are not even trained programmers) still do, even though spreadsheet packages support more user-friendly cell addressing. (In Excel, for example, this feature is called the "name manager".)

We think the challenge lies in learning and adoption of such features, and there are design challenges for doing so, as was demonstrated in Calculation View [54]. Approaches such as natural language cell addressing (e.g., [28]) might also help spreadsheet users both during authoring and comprehension activities, just as good variable names help professional programmers. We believe that solving the problem of semantic cell addressing is a crucial step towards visions such as end-user software engineering [5] in the spreadsheet domain.

Our results also revealed that under-the-hood comprehension was broader than formulas. We found that 4 out of 15 participants referred to the numeric/string formats applied to cells, conditional formatting rules and the data validations in their spreadsheets: these included instances of spreadsheet developers seeking to modify and reuse these contents, as well as end-users seeking to simply use the spreadsheet for a task (e.g., P01's data entry required knowing the correct format / validation rule for entering date in a cell). We think these under-the-hood aspects are under supported and our results reveal opportunities for improving the visibility of these items for the end-user, and making them reusable for developers (e.g., to reuse data validations from one spreadsheet to another).

Finally, deviating from their current task to go on information-seeking tangents accounted for a surprisingly high fraction of comprehension activities: an average of 40% of participants' time went into information seeking, and having to regain context—likely incurring a cognitive burden—rather than just reading sequentially and understanding. Such information seeking tangents also feature in program comprehension during maintenance activities [37, 31], where information seeking activities account for as much as 50% of programmers' time during maintenance activities. To reduce this overhead, software engineering researchers have adopted HCI theories such as cognitive dimensions of notations [3], information foraging [49], minimalistic learning [9] and attention investment [4] to provide relevant information to programmers in ways that will ease their information seeking and comprehension. Our findings reveal the opportunity for investigating the applicability of these theories, and leveraging them, in the spreadsheet domain (both for spreadsheet developers, as well as for end users).

7.2 Implications: the Author Side of the Equation

On the author side, recall that spreadsheet comprehension difficulties (such as finding information about the context) were *not because* of the lack of effort on the author's part in design, formatting or layout—they are *despite* such efforts. We found evidence of 1) authors adding in formatting and layout, but missing legends for what they mean, 2) documentation being at the end of the

spreadsheet data (rather than at the beginning), hence potentially missed or noticed too late by the reader, and 3) documentation remained elsewhere (e.g., emails). Prior research also shows that spreadsheet users spend efforts to make spreadsheets more comprehensible (e.g., good layout, documentation, format) [58] but omit documenting contextual details in their spreadsheet (e.g., [18, 39]).

If the authors are making reasonable efforts, and still the efforts seem incomplete for the reader, then the problem might be with the tools. Specifically, we believe that the grid nature of spreadsheets, or the need to go out of context to click to write comments might be ill-suited to the documentation needs and workflows of users, and [58] reported that as high as 20% of spreadsheet authors documented their spreadsheets outside of the spreadsheet, rather than as part of it. We need further research into authoring processes in spreadsheets to see how writing documentation fits into authors' workflows.

A concrete example of this is that English language spreadsheets are typically authored from left to right, top to bottom. If the author adds documentation at the very end of an authoring session, the least-effort location for the documentation to be placed is to the right or at the bottom of the spreadsheet, where there is ample empty grid space, as opposed to trying to create space in the top left, which may involve inserting rows and columns, that can break carefully planned formula references and layouts. The problem is that English language spreadsheets are also typically *read* from left to right, top to bottom. Placing the documentation in the bottom right makes it at best inconvenient for the reader to have to find and navigate to the documentation, and at worst makes it possible that the reader fails to find and read the documentation altogether. We found such instances of documentation available at the end of the spreadsheet, as well as outside it. So, if spreadsheet authors document after authoring in a manner that is more ad hoc than systematic, we need to be able to support those workflows and behaviors, while still making the documentation available in context for the reader.

Further, based on evidence of people spending too much time trying to format their spreadsheets in our tasks, we suspect that there might be various tradeoffs in the spreadsheet authoring process. Some possibilities are that: 1) the effort put into formatting and layout alone might be so high that the author might not have the time to create additional written documentation, 2) under time constraints, spreadsheet authors might focus on getting things done, rather than making the layout of their spreadsheets better, or 3) the content and layout that best suits one task might hinder comprehension during other tasks. For example, including intermediate results of computations in the grid might be helpful for debugging a formula, but add to clutter for decision making.

Tradeoffs such as these are fundamental to various human activities at work and have been reported in other domains such as web search [50]. Theories such as attention investment [4] and information foraging [50] offer fundamental explanations for why these constraints occur. Research has begun exploring theoretical models to address these tradeoffs as well as considered operationalizing them in some domains (e.g., [50]) and their results are transferable to spreadsheet tools. For example, minimizing the costs of formatting spreadsheets (e.g., by automatically formatting them for users) could help users spend the remaining time or attention in more important tasks like documenting context. Another option is to also bring down the costs of documenting context (e.g., by generating legends of formats), that might otherwise take up too much time and attention of users. However, open problems remain both in the development and operationalization of theoretical models, as well as in transferring the theoretical predictions and explanations into design options in tools.

7.3 Call for Action: Spreadsheet Comprehension Woes Are A Potential Threat To Accuracy.

Finally, going beyond the difficulties involved in reading spreadsheets, and in making them readable, our results offer novel insights into spreadsheet errors—namely that some of them are due to misreading or miscomprehension.

It is well known in the spreadsheet research community that spreadsheet errors are a major problem and that studies of spreadsheet audits reveal errors in a very high fraction of them [45]. Although theoretical taxonomies [12] mention that errors might occur in all stages of human activities, hence also in spreadsheet reading and interpretation, empirical evidence for their occurrence and causes is scarce. Prior to our work, the state of the art was data from Caulkins et al.'s study [6] which found that a high fraction of managers reported misinterpreting spreadsheets. Our results build on top of their mention of spreadsheet comprehension errors and offer evidence that spreadsheet comprehension and information seeking difficulties might be at the root of other kinds of spreadsheet errors, such as incorrect formulas or incomplete data entry.

First, participants in our study ended up making errors because of the lack of information available in time and having to constantly switch contexts between doing their task and seeking information (e.g., when looking up range references across sheets to author formulas). Participants faced difficulties due to the *hidden data problem*, suggesting that it might be easy to miss content because of the unbounded size of the grid. We also found evidence of one participant (P07) overlooking a second sheet in the spreadsheet because of being overwhelmed by it, and as a result he did not complete his expense entry task. These are in fact difficulties rooted in failures of comprehension and information seeking, but errors at this stage are largely unobserved, and manifest only later during data entry or formula editing. By tracing the source of the error upstream to the comprehension stage rather than fixate on its downstream effects, we are better positioned to prevent such errors through tool support, rather than merely to detect them and try to help the user recover after they have been made.

Second, 12 out of 15 participants (exceptions: P02, P09, P14) wanted to go back to the spreadsheet's author to seek clarifications about their comprehension of the spreadsheet. However, the author might not be readily available, or may be completely unavailable. They might have left the organization or changed role, which prior work has identified as a common reason for a spreadsheet to be transferred from one person to another [22]. Thus, the user might have no option but to work with their guesses, and incorrect guesses cause errors.

Third, even when a user can verify their hypotheses with the spreadsheet author, there is evidence from sensemaking and theories of human working memory [49] that they might miss confirming some of their hypotheses, and their reasoning for their beliefs. In our study, only three participants (P04, P11, P15) kept notes of their tentative understandings that they needed to verify with the author and such misses could result in confirmation bias, that could potentially lead to critical errors during decision making. While some research (e.g., [20]) exists on guarding against cognitive biases, there is opportunity for design research in this area, especially since users are often overconfident about the correctness of their spreadsheets [46].

Finally, prior research suggests that spreadsheets are frequently used under time pressure [39]. As suggested by research in human factors (e.g., [52]), time pressure could compound the difficulties of comprehending spreadsheets and potentially increase the likelihood of users committing errors.

In summary, we have provided the most comprehensive and detailed evidence to date that spreadsheet comprehension is an important source of spreadsheet errors, despite users tending not to view spreadsheet comprehension as a distinct activity [6]. There is a clear and urgent need for tool design to better support users when comprehending spreadsheets, and for design research to address the follow-up questions raised by our observations.

8 CONCLUSION

Hundreds of millions of spreadsheet users [56] read spreadsheets: to make critical decisions, enhance and reuse their spreadsheets, or to enter data. Some prior studies suggest that spreadsheet comprehension can be tedious [18] and even error-prone, with as many as 25% of participants in a study reporting spreadsheet miscomprehension [6]. Yet, the disproportionate lack of research and commercial interest in *reading* spreadsheets—the overwhelming focus has been on *writing* spreadsheets—has, until now, left a gap in our understanding. This limited our ability to support millions of users in their spreadsheet comprehension activities.

In this paper, we begin to bridge the gap in our knowledge about spreadsheet readers. We conducted the first study observing users comprehending spreadsheets as part of their actual work or personal tasks. By coding the think-aloud verbalizations and screen actions of participants (N=15) in 20-second intervals, we gained a fine-grained view of users' spreadsheet comprehension activities. Our novel study design helped reveal new insights into spreadsheet users' information needs, strategies, and barriers during comprehension. Our key findings are that:

- *Spreadsheet comprehension is more than just formula comprehension.* In contrast to prior work that has largely focused on comprehending formulas that lie under-the-hood, participants in our study needed to comprehend what was over-the-hood (e.g., data, charts) and had difficulties dealing with large amounts of data and in seeking out how far the spreadsheet's contents extended on the unbounded grid. Even under-the-hood comprehension involved more than just formulas (e.g., data validations, conditional formatting, numeric and string formats).
- *Spreadsheet comprehension involves information-seeking detours up to 40% of the time.* Participants spent only 60% of the time reading the contents of the spreadsheet, such as labels, data, formulas, and documentation, directly. Even without accounting for the time spent looking up formula references, participants spent as much as 40% of their time going on information-seeking tangents to gather critical information needed for their comprehension—frequently focusing away from their current activity, navigating to a different location and then spending additional time and cognitive effort finding their way back and regaining lost context.
- *Spreadsheet comprehension strategies involve guessing and going back to the author.* To deal with the difficulties of straightforward comprehension and sensemaking, participants often adopted a hypothesis testing strategy. In turn, the same difficulties in seeking and understanding information hindered this strategy. As a result, participants either went back to the author, hindering task progress, or worse, pressed on with their tentative hypotheses—at best informed guesses—to make progress in their tasks, creating the risk of error.

This is the first think-aloud study of spreadsheet comprehension. Fine-grained data on comprehension activities as they unfold in real time allows us to derive findings beyond what was possible to glean from prior interview and survey studies [6] [15]. Our findings reveal new opportunities for design research, including in: 1) the application of existing theories (e.g., cognitive dimensions of notations, information foraging) to support spreadsheet comprehension, 2) the application of theories such as minimal learning and attention investment, to help authors make their spreadsheets readable in ways that will minimize their effort while still capturing critical context, and 3) designing tools and interfaces that will make visible what is hidden in spreadsheets—both under-the-hood as well as over—in ways that benefit the millions of users who depend on spreadsheets every day.

ACKNOWLEDGMENTS

The authors would like to thank the study participants for agreeing to bring their work tasks to the study. They also thank the anonymous reviewers for their thought-provoking questions. Last but not the least, they thank their colleagues at Microsoft Research, Cambridge for their valuable inputs on study design.

REFERENCES

[1] Daniel Ballinger, Robert Biddle, and James Noble. "Spreadsheet Visualization to improve end-user understanding." In Proceedings of the Asia-Pacific symposium on Information Visualization-Volume 24, pp. 99-109. 2003.
[2] Kenneth R Baker, Lynn Foster-Johnson, Barry Lawson, and Stephen G. Powell. "A survey of MBA spreadsheet users." Spreadsheet Engineering Research Project. Tuck School of Business 9 (2006).
[3] Alan Blackwell, and Thomas Green. "Notational systems–the cognitive dimensions of notations framework." HCI models, theories, and frameworks: toward an interdisciplinary science. Morgan Kaufmann (2003).
[4] Alan F. Blackwell "First steps in programming: A rationale for attention investment models." In Proceedings IEEE 2002 Symposia on Human Centric Computing Languages and Environments, pp. 2-10. IEEE, 2002.
[5] Margaret Burnett, Brad Myers, Mary Beth Rosson, and Susan Wiedenbeck. "The next step: from end-user programming to end-user software engineering." In CHI'06 Extended Abstracts on Human Factors in Computing Systems, pp. 1699-1702. 2006.

[6] Jonathan P. Caulkins, Erica Layne Morrison, and Timothy Weidemann. "Spreadsheet errors and decision making: Evidence from field interviews." Journal of Organizational and End User Computing (JOEUC) 19, no. 3 (2007): 1-23.

[7] John L. Campbell, Charles Quincy, Jordan Osserman, and Ove K. Pedersen. "Coding in-depth semi structured interviews: Problems of unitization and intercoder reliability and agreement." Sociological Methods & Research 42, no. 3 (2013): 294-320.

[8] Diogo Canteiro, and Jácome Cunha. "SpreadsheetDoc: An Excel Add-in for Documenting Spreadsheets." In Proceedings of the 6th National Symposium of Informatics. 2015.

[9] John M. Carroll, and Mary Beth Rosson. "Paradox of the active user." In Interfacing thought: Cognitive aspects of human-computer interaction, pp. 80-111. 1987.

[10] Chris Chambers and Chris Scaffidi. "Struggling to excel: A field study of challenges faced by spreadsheet users." In 2010 IEEE Symposium on Visual Languages and Human-Centric Computing, pp. 187-194. IEEE, 2010.

[11] Kerry Shih-Ping Chang, and Brad A. Myers. "Using and exploring hierarchical data in spreadsheets." In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, pp. 2497-2507. 2016.

[12] Elaine Dobell, Sebastian Herold, and Jim Buckley. "Spreadsheet Error Types and Their Prevalence in a Healthcare Context." Journal of Organizational and End User Computing (JOEUC) 30, no. 2 (2018): 20-42.

[13] The FAST Standard. https://www.fast-standard.org/the-fast-standard/

[14] David J Gilmore and Thomas R. G. Green. "Comprehension and recall of miniature programs." International Journal of Man-Machine Studies 21.1 (1984): 31-48.

[15] Valentina Grigoreanu, Margaret Burnett, Susan Wiedenbeck, Jill Cao, Kyle Rector, and Irwin Kwan. "End-user debugging strategies: A sensemaking perspective." ACM Transactions on Computer-Human Interaction (TOCHI) 19, no. 1 (2012): 1-28.

[16] Sandra G. Hart and Lowell E. Staveland. "Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research." In Advances in psychology, vol. 52, pp. 139-183. North-Holland, 1988.

[17] Karin Hodnigg and Roland T. Mittermeir"Metrics-Based Spreadsheet Visualization: Support for Focused Maintenance".Proceedings of the. European Spreadsheet Risks International Group (EuSpRIG) 2008.

[18] David G Hendry, and Thomas RG Green. "Creating, comprehending and explaining spreadsheets: a cognitive interpretation of what discretionary users think of the spreadsheet model." International Journal of Human-Computer Studies 40, no. 6 (1994): 1033-1065.

[19] David G. Hendry, and Thomas R. G. Green. "CogMap: a visual description language for spreadsheets." Journal of Visual Languages & Computing 4, no. 1 (1993): 35-54.

[20] Nitesh Goyal, and Susan R. Fussell. "Effects of sensemaking translucence on distributed collaborative analysis." In Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing, pp. 288-302. 2016.

[21] Thomas A. Grossman and Özgür Özlük. "A paradigm for spreadsheet engineering methodologies." Proceedings of EuSpRIG 2004 Conference Risk Reduction in End User Computing: Best practice for spreadsheet users in the new Europe (EuSPRiG). 2004.

[22] Felienne Hermans, Martin Pinzger, and Arie Van Deursen. "Supporting professional spreadsheet users by generating leveled dataflow diagrams." Proceedings of the 33rd International Conference on Software Engineering. 2011.

[23] Felienne Hermans, Martin Pinzger, and Arie van Deursen. "Detecting code smells in spreadsheet formulas." In 2012 28th IEEE International Conference on Software Maintenance (ICSM), pp. 409-418. IEEE, 2012.

[24] Felienne Hermans, and Danny Dig. "Bumblebee: a refactoring environment for spreadsheet formulas." In Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering, pp. 747-750. 2014.

[25] Jean-Michel Hoc. Cognitive psychology of planning. Academic Press Professional, Inc., 1988.

[26] Takeo Igarashi, Jock D. Mackinlay, Bay-Wei Chang, and Polle T. Zellweger. "Fluid visualization of spreadsheet structures." In Proceedings. 1998 IEEE Symposium on Visual Languages (Cat. No. 98TB100254), pp. 118-125. IEEE, 1998.

[27] Cory Kissinger, Margaret Burnett, Simone Stumpf, Neeraja Subrahmaniyan, Laura Beckwith, Sherry Yang, and Mary Beth Rosson. "Supporting end-user debugging: what do users want to know?." In Proceedings of the working conference on Advanced visual interfaces, pp. 135-142. 2006.

[28] Krassimira B. Ivanova, Koen Vanhoof, Krassimir Markov, and Vitalii Velychko. "Introduction on the Natural Language Addressing." International Journal of Information Technologies and Knowledge 7, no. 2 (2013): 139-146.

[29] Nima Joharizadeh, Advait Sarkar, Andrew D. Gordon, and Jack Williams. "Gridlets: Reusing spreadsheet grids." In Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems, pp. 1-7. 2020.

[30] Bennett Kankuzi, and Yirsaw Ayalew. "An end-user oriented graph-based visualization for spreadsheets." In Proceedings of the 4th international workshop on End-user software engineering, pp. 86-90. 2008.

[31] Amy J. Ko, and Brad A. Myers. "Designing the whyline: a debugging interface for asking questions about program behavior." In Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 151-158. 2004.

[32] Andrea Kohlhase, Michael Kohlhase, and Ana Guseva. "Context in Spreadsheet Comprehension." In SEMS@ ICSE, pp. 21-27. 2015.

[33] Andrea Kohlhase, and Alexandru Toader. "FEncy: Spreadsheet Formulae Exploration." In CICM Workshops. 2014.

[34] Andrea Kohlhase. and Michael Kohlhase, 2009, October. Semantic transparency in user assistance systems. In Proceedings of the 27th ACM international conference on Design of communication (pp. 89-96).

[35] Thomas D Latoza, David Garlan, James D. Herbsleb, and Brad A. Myers. "Program comprehension as fact finding." In Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering, pp. 361-370. 2007.

[36] Joseph Lawrance, Robin Abraham, Margaret Burnett, and Martin Erwig. "Sharing reasoning about faults in spreadsheets: An empirical study." In Visual Languages and Human-Centric Computing (VL/HCC'06), pp. 35-42. IEEE, 2006.

[37] Joseph Lawrance, Christopher Bogart, Margaret Burnett, Rachel Bellamy, Kyle Rector, and Scott D. Fleming. "How programmers debug, revisited: An information foraging theory perspective." IEEE Transactions on Software Engineering 39, no. 2 (2010): 197-215.

[38] Walid Maalej, Rebecca Tiarks, Tobias Roehm, and Rainer Koschke. "On the comprehension of program comprehension." ACM Transactions on Software Engineering and Methodology (TOSEM) 23, no. 4 (2014): 1-37.

[39] Mukul Madahar, Pat Cleary, David Ball. "Categorisation of Spreadsheet Use within Organisations, Incorporating Risk: A Progress Report". Proceedings of the European Spreadsheet Risks Interest Group (EuSPRIG). 2007 37-45 ISBN 978-905617-58-6.

[40] Modano. https://www.modano.com/guidelines.

[41] Brad A. Myers Margaret M. Burnett, Susan Wiedenbeck, and Amy J. Ko. "End user software engineering: CHI 2007 special interest group meeting." In CHI'07 extended abstracts on human factors in computing systems, pp. 2125-2128. 2007.

[42] Bonnie A. Nardi, and James R. Miller. "An ethnographic study of distributed problem solving in spreadsheet development." In Proceedings of the 1990 ACM conference on Computer-supported cooperative work, pp. 197-208. 1990.

[43] Bonnie A. Nardi, "A small matter of programming: perspectives on end user computing". MIT press, 1993.

[44] Raquel Navarro-Prieto and Jose J. Cañas. "Are visual programming languages better? The role of imagery in program comprehension." International Journal of Human-Computer Studies 54, no. 6 (2001): 799-829.

[45] Raymond R Panko,. "Two corpuses of spreadsheet errors." In Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, pp. 8-pp. IEEE, 2000.

[46] Panko, Raymond R. "Spreadsheet errors: What we know. what we think we can do." arXiv preprint arXiv:0802.3457 (2008).

[47] Alexandre Perez, and Rui Abreu. "A diagnosis-based approach to software comprehension." Proceedings of the 22nd International Conference on Program Comprehension. 2014.

[48] David Piorkowski, Austin Z. Henley, Tahmid Nabi, Scott D. Fleming, Christopher Scaffidi, and Margaret Burnett. "Foraging and navigations, fundamentally: developers' predictions of value and cost." In Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, pp. 97-108. 2016.

[49] Peter Pirolli and Stuart Card. "The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis." In Proceedings of international conference on intelligence analysis, vol. 5, pp. 2-4. 2005.

[50] Peter Pirolli. Information foraging theory: Adaptive interaction with information. Oxford University Press, 2007.

[51] John F Raffensperger. "New guidelines for spreadsheets." Proc. European Spreadsheet Risks Int. Grp. (EuSpRIG). 2001.

[52] James Reason, Human error. Cambridge university press, 1990.

[53] Sohon Roy, Felienne Hermans, and Arie van Deursen. "Spreadsheet testing in practice." In 2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER), pp. 338-348. IEEE, 2017.

[54] Advait Sarkar, Andrew D. Gordon, Simon Peyton Jones, and Neil Toronto. "Calculation view: multiple-representation editing in spreadsheets." In 2018 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), pp. 85-93. IEEE, 2018.

[55] Justin Smith, Justin A. Middleton, and Nicholas A. Kraft. "Spreadsheet practices and challenges in a large multinational conglomerate." In 2017 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), pp. 155-163. IEEE, 2017.

[56] Christopher Scaffidi, Mary Shaw, and Brad Myers. "Estimating the numbers of end users and end user programmers." 2005 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05). IEEE, 2005.

[57] Christopher Scaffidi, Joel Brandt, Margaret Burnett, Andrew Dove, and Brad Myers. "SIG: end-user programming." In CHI'12 Extended Abstracts on Human Factors in Computing Systems, pp. 1193-1996. 2012.

[58] SERP survey results. http://faculty.tuck.dartmouth.edu/images/uploads/faculty/serp/serp_results.pdf. Retrieved 14 September, 2020.

[59] Hidekazu Shiozawa, Ken-ichi Okada, and Yutaka Matsushita. "3d interactive visualization for inter-cell dependencies of spreadsheets." In Proceedings 1999 IEEE Symposium on Information Visualization (InfoVis' 99), pp. 79-82. IEEE, 1999.
[60] Elliott Soloway, and Kate Ehrlich. "Empirical studies of programming knowledge." IEEE Transactions on software engineering 5 (1984): 595-609.
[61] Margaret-Anne Storey. "Theories, tools and research methods in program comprehension: past, present and future." Software Quality Journal 14, no. 3 (2006): 187-208.
[62] Al-Corbin Strauss. "Open coding" in "Basics of qualitative research". London: Sage: 61-74. 1990
[63] Anneliese Von Mayrhauser and A. Marie Vans. "From program comprehension to tool requirements for an industrial environment." IEEE Second Workshop on Program Comprehension (1993): 78-86
[64] Aaron Wilson, Margaret Burnett, Laura Beckwith, Orion Granatir, Ledah Casburn, Curtis Cook, Mike Durham, and Gregg Rothermel. "Harnessing curiosity to increase correctness in end-user programming." In Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 305-312. 2003.
[65] 20 Principles for Good Spreadsheet Practices. https://learn.filtered.com/blog/20-principles-for-good-spreadsheet-practice
[66] Melissa Rodriguez Zynda. "The first killer app: A history of spreadsheets." interactions 20, no. 5 (2013): 68-72.

A APPENDIX A: CODE BOOK

Summary of data	
CODES	Count
No. of information needs codes	25
No. of strategies codes	26
No. of barriers codes	19
Total no. of codes	71

COMPREHENSION AND INFORMATION SEEKING ACTIVITIES			
Themes / categories	**Code name**	**Code definition**	**Relation to prior work**
Read (directly, without further info. seeking detour), over the hood	understand	Read and interpret the contents in the grid (verbalizations of where, meaning, etc. belong here; only info. seeking uses other codes) *	
	chart	Read and interpret chart directly	
	extent	How far does the spreadsheet extend? Are there hidden cells?	organization [Kohlhase, 2015]
	check data	Read the data directly and check it (against judgment, or against another source)	evaluation [Kohlhase, 2015]
	check: others	Other kinds of checking (e.g., does the link work)	evaluation [Kohlhase, 2015]
Read as is, under the hood	formula	Read and interpret formulas directly (includes looking up references and cell labels)	Same as formula [Kohlhase, 2015]
	check formula	Check formula correctness (by understanding it, or by testing values)	evaluation [Kohlhase, 2015]
Questions about behavior & contents within spreadsheet	format details	How is something formatted? (special case of how to)	How Goal [Kissinger et al. 2006]
	debug	Investigate the source of an error (or a suspected error)	Determine if error exists, identify missing code [von Mayerhauser, 1993]
	why this value?	Why is a value what it is (and not the other one, or not what I expect it should be)	reason [Kohlhase, 2015]
	difference	What is the difference between two items in the spreadsheet?	
	dependents	Dependents of a cell (where is this used? If I change this, will it affect others)	provenance, usage, purpose in [Kohlhase, 2015] or oracle/formula [Kissinger, 2006], function call graph [von Mayrhauser, 2003]
	where	Locating something with the spreadsheet (including formula /chart precedents)	layout/organization [vonMayerhauser, 1993], organization, provenance [Kohlhase, 2015],
	what happened to x?	Look for something that I expect to be in the spreadsheet (more vague / generic goal than where or how)	organization [Kohlhase, 2015]
	data statistics	Seek aggregate statistics for data, rather than dealing with raw data	
Context	Rationale	Context questions of the form "why is...?"	reason + purpose [Kohlhase, 2015]
	Meaning	Context questions of the form "what does this mean?"	concept [Kissinger et al., 2006], definition [Kohlhase, 2015]. Also overlaps with definition of terms/concepts and acronym definitions [von Mayrhauser, 1993]
	Source	Context questions pertaining to the external source of the data in the spreadsheet	provenance[Kohlhase, 2015]
	History	Context questions of the form "what has changed"?	History [Kohlhase, 2015]
Unfamiliar concepts	Syntax	What does this function / syntax mean?	Concept[Kissinger, 2006], feature challenges [Smith,2017]
	Error codes / messages	What does this error code / message mean?	
	Howto	How can I do X, or How did X achieve this?	How Goal [Kissinger et al., 2006], feature challenges [Smith, 2017
	Whats this?	Something unexpected--what is that?	whoa?! [Kissinger et al., 2006], feature challenges [Smith, 2017]
Other codes	Find file	Find what is in a workbook / look for a specific workbook	
	Non comprehension (non. compr)	Non-comprehension activities (e.g., talking to researcher, editing)	

STRATEGIES			
Themes / categories	**Code name**	**Code definition**	**Relation to prior work *****
What author put in	label	Everytime participant attended to labels	
	documentation	Everytime participant attended to some form of documentation (not just labels)	
	formatting	Explicit mentions of formatting	
	convention	Explicit mentions of conventions	
	Data validations	Data validations added by author helped comprehend what to do	
	Filename	Attending to file name (special case of label)	
What tool offered + what reader used	Cues	Attending to cues provided by tool (current row/column highlight, cell reference highlight)	
	status bar	Data summary that appears at the bottom right of the spreadsheet tool	
	Arrows	The cell precedent / dependent arrow feature in Excel	Related to function call graph [von Mayrhauser, 2003]
	subformula evaluation	Formula partial evaluation that appears on F9 (on formula bar)	evaluation [Kohlhase, 2015]
What participants did	Ordering	Reoder the spreadsheet to make it easier to read	
	Visualize	Create and use charts, rather than read data	
	Visibility	Make parts of spreadsheet visible or invisible to make reading easier (freeze panes, zoom)	
Info. sources	Help	Use In-tool help / tool-tips	Help [Kissinger et al., 2006]
	web	Web search to seek information	
Other behaviors and strategies	Comparison	Deduce something based on the similarity / difference with other things	
	Memory	Recall from memory of prior knowledge / what was already seen	Situation knowledge [von Mayrhauser, 1993]
	Reconnaissance	Navigate away and back to seek something	
	Hypothesis	Tentative conclusions about spreadsheet's working rather than certain statements	Strategy hypothesis [Kissinger et al., 2006], hypothesis [Grigoreanu et al., 2012], Generate or revise hypothesis [vonMayerhauser, 1993]
	Author	References to author (than spreadsheet)	Related to Explanation [Kissinger, 2006]: one participant in their study explained something to another participant, when working in pairs.
	References	Explicitly reading out references, rather than concepts in domain	
Navigation strategies	Systematic reading (sys. read)	Reading something systematically	Related to "determine which program segment to next examine" [vonMayerhauser, 1993]
	Random	Explicit mentions of randomness in doing something (different from skimming, that we did not explicitly code)	
	Targeted	Read something targeted (just non-zero values)	
	Block	Read one cell in a block and then check if the entire block has the same formula	general classification of functions, so that if one is understood, the rest in the group is also understood [von Mayerhauser, 1993]
Making edits	edit: readability	Make edits intended towards making something readable (e.g., color, format)	
	edit: others	Other kinds of edits (e.g., notes to author, task-related edits, data input)	

BARRIERS			
Themes / categories	**Code name**	**Code definition**	**Relation to prior work *****
Reader-related	Limited time	Time constraints	Time/Budget challenges [Smith, 2017]
	unfamiliar lang.	Unfamiliar language	
	Limited (lim) domain experience (exp)	Explicit mention of limited domain knowledge / expertise	People challenges [Smith, 2017]
	Limited(lim) spreadsheet experience (exp)	Limited experience / knowledge of spreadsheet concepts / techniques	People challenges, manual tasks [Smith, 2017]
	fiddly	fiddly interaction (mislicking & misscrolling, other motor errors)	
Author-caused	messy	Explicit mention of messy spreadsheet, cluttered with unwanted things (e.g., "to be deleted")	
	Missing data	Missing data required to infer something	
	No formula	Spreadsheet has calculated value instead of formulas	Manual tasks [Smith, 2017]
	Poor formatting	Unclear, ambiguous or undocumented arbitary formatting / layout	
	poor docs	Insufficient, unclear or unavailable documentation	Hendry and Green, 1994
Other	Overwhelmed	Explicit mentions of being overwhelmed / explicit sigh	whoa?! [Kissinger et al., 2006]
Tool limitations	Cross-sheet references (ref)	Explicit mention of cross sheet difficulties	[Hendry and Green, 1994], Complexity challenges [Smith,2017]
	Long calculation (calc) chain	Explicit mention of long calculation chain problem	Hendry and Green, 1994], Complexity challenges [Smith, 2017]
	Long formula	Explicit mention of long and complex formula	[Hendry and Green, 1994], Complexity challenges [Smith,2017]
	Hidden data validations	Hidden data validations	
	Limited (lim.) debugging support	Limited debugging support in the formula bar (e.g., debugging / see what something evaluated to)	
	Lim. screen space	Limited screen space causing difficulties seeing multiple things / moving things around	Infrastructure limitations [Smith, 2017]
	unclear text	Unclear error message or help text (the text is provided by the tool and not put in by the user or author)	Help resource challenges [Smith, 2017]
	toolsettings	Missing tool customizations; inconsistent keymaps across OS	Config [Chambers,2012]

Rows in grey represent codes that have a direct corresponding code in prior work. The other codes either disambiguate codes in prior work, or only overlap in some aspect, but are not exact matches.

*Note that codes from [Kohlhase, 2015] pertain to various descriptors that authors use to describe their spreadsheet. They are not really information needs of users; hence, though the codes might be similar, our coding scheme was different. If a participant made a statement about what they read / interpret on the spreadsheet, or verbalize the definition / meaning of an abbreviation, we coded it as the "understand" activitiy, rather than the descriptors uthors describe their spreadsheets.

** Several codes from prior studies overlapped with codes in our study (e.g., provenance in [Kohlhase, 2015] overlaps with our where and source); but we needed to disambiguate codes based on what exactly participants were looking for, where, and why in their broad range of tasks.

***Also note that while several information needs overlap with prior observations, strategies and barriers have been largely overlooked in prior literature

**** [Smith et al., 2017] includes spreadsheet comprehension itself as a challenge, but some of the other challenges mentioned here also overlap with spreadsheet difficulties.