University of Twente

The Influence of Code Complexity on Review Efficiency, Effectiveness and Workload in Embedded Software Development

Master Thesis

Human Factors & Engineering Psychology

Student: Gina Jahncke

Student number: s 3052206

1st Internal Supervisor:

Martin Schmettow

2nd Internal Supervisor:

Dr. Funda Yildirim

External Supervisor:

Jonas Wolf

Date: 19.07.2023

Table of Contents

1 Abstract

Code review plays a crucial role in ensuring the quality and reliability of software. However, its effectiveness depends on the cognitive abilities and performance of individual programmers. Drawing upon the theoretical framework of human factors psychology, this study investigates the impact of code complexity on code review workload, efficiency, and effectiveness. An experiment was conducted involving developers from a software development company, Vector Informatik GmbH, who reviewed code samples of varying complexity levels. Subjective workload ratings, review times, and defect detection rates were examined to test the influence of two common code complexity metrics, i.e., cyclomatic complexity and nesting depth on these outcomes. The findings indicate that higher levels of code complexity are associated with elevated workload, increased review times, and decreased fault detection rates. If these results turn out to be true, they suggest that code complexity poses challenges for reviewers in comprehending and maintaining complex code, potentially hindering effective code review. By applying insights from human factors psychology, this study emphasises the cognitive challenges associated with code complexity and highlights the need for strategies to mitigate its negative effects on code review.

Keywords: *code review, code complexity, mental workload, performance, defect detection, human factors*

2 Introduction

In the early 2000s, several Toyota vehicles were involved in a series of accidents caused by unintended acceleration. Without driver input, the vehicles suddenly accelerated, sometimes leading to serious injuries and loss of life. Consequent investigation revealed that the source of the issue was a software bug in the electronic throttle control system (ETCS). This bug resulted in the system misinterpreting signals from the accelerator pedal, causing an unintended acceleration. After a lengthy search for a cause, the origin of the problem was attributed to inadequate code review practices during the development of the ETCS software. The review process failed to identify the defect, resulting in significant safety risks for drivers and passengers (Barr, 2013).

This incident highlights the importance of effective code review in software development, especially when dealing with safety-critical systems. Code review enables developers to identify problems early in the development process and provides an opportunity resolve issues before software is deployed in real-world settings (Bavota & Russo, 2015). Reviewers thereby ensure that a given code performs as intended, potentially preventing catastrophic consequences such as those experienced by Toyota.

The effectiveness of code review is therefore contingent upon the performance of individual coders. In code review, individual programmers must comprehend and resolve problems, rendering it an intellectual task that heavily depends on human cognition (Huang et al., 2015). Therefore, human factors psychology provides a theoretical and methodological framework for understanding the cognitive and perceptual processes involved in this task. It explores how human capabilities and limitations affect human perception and performance in various situations and interactions (Wickens et al., 2021). In the context of code review, it is the software developer that interacts with the code under review.

One of the factors that appears to influence this interaction is the complexity of code. Code complexity is an integral concept in software engineering and describes the level difficulty involved in comprehending and maintaining source code (Curtis et al., 1979). Summarised by numerous metrics, it encompasses various factors such as code size, structural complexity, and many more (Nuñez-Varela et al., 2017). If such complexity levels are high, they may negatively impact software quality by making it difficult for reviewers to understand and consequently maintain code (Bacchelli & Bird, 2013).

Yet, there is limited research on the effects of code complexity on review workload and performance. The current study aims to fill this gap and investigates the impact of code

complexity on code review workload, efficiency, and effectiveness. To test this impact, an experiment in which participants reviewed code samples of varying levels of complexity was conducted. Subjective workload ratings, review times, and fault detection rates were measured to evaluate the effect of code complexity on these outcomes.

2.1 Code review

Software development is a complex process, involving several phases such as planning, design, coding, testing, and maintenance of software. An important step in this process is code review. First formally introduced as “code inspection” by Fagan (1976), it involved a highly structured review process that included planning, preparation, review, and follow-up meetings. However, in recent years, this type of formal inspection was largely replaced by a modern, more practicable process, that is suitable for the iterative nature of software development (Bacchelli & Bird, 2013; Stein et al., 1997). Modern code review now relies more heavily on tools and involves a more informal process where developers collaborate asynchronously on smaller code changes (Rigby & Bird, 2013; Sadowski et al., 2018).

The primary purpose of code review is to identify defects before a software is deployed. A survey by Bacchelli & Bird (2013) showed that the majority of managers as well as developers consider the identification of bugs as one of its central goals. In fact, almost half of the managers as well as programmers considered it the number one reason for code review. This is not unjustified. Several studies have shown that poorly reviewed code has a detrimental impact on software quality. For example, Bavota & Russo (2015) found that unreviewed code is over two times more likely to introduce bugs than reviewed code. Correspondingly, a case study of various projects estimated that inadequate code review coverage can result in the production of components with up to two additional post-release defects (McIntosh et al., 2014).

Thus, by identifying shortcomings, reviewers ensure that software code is of high quality and thereby ensure the safety of a system. Ensuring software quality in this way is especially important in embedded software development. In embedded software systems, the software interacts directly with hardware. This has two consequences. Due to its embedded nature, faults in embedded systems are less observable and therefore difficult to detect once the system is employed. Additionally, any undetected error can cause a significant impact on the system's functionality and safety, meaning failure can result in significant financial and personal loss, including loss of life, as seen in various related accidents (Barr, 2013; Leveson, 2011). Therefore, in embedded software projects, code review is particularly crucial due to the

critical nature of the systems they control. It is imperative to detect and fix errors, including unhandled conditions, effectively to avoid fatal consequences.

2.2 Human factors in code review

Despite the apparent importance of code review, its effectiveness seems to be deficient. A recent study by Khoshnoud et al. (2022) has shown that there is a considerable amount of left-over defects, even after a code has been reviewed. After manually examining pull requests from 77 open-source Github projects, they discovered that in 173 pull requests, at least 187 defects were missed. A taxonomy of the identified defects can be found in their paper. These figures raise concerns about the ability of reviewers to detect errors and highlight the importance of enhancing review effectiveness. Consequently, there is a spreading interest in understanding the role of human factors in code review.

Despite the increasing automation in software development, code review remains a predominantly manual process, that is involving human beings (e.g. Sadowski et al., 2018). The review process requires programmers to read code, understand its functionality and behaviour, and resolve potential errors. Given that this task relies on human cognition, its effectiveness depends on the performance of the individuals involved. In fact, industrial data suggests that 87 percent of residual software defects can be attributed to individual cognitive failure (Huang et al., 2015). This is likely to be particularly relevant for modern code review, where review quality is dependent on single coders rather than a team. Consequently, modern code review is especially susceptible to human error.

Previous research has demonstrated human factors to be a major contributor to code review quality. In software engineering, human error may originate from various sources (Anu et al., 2018; Huang et al., 2012; Reason, 1990). Among the factors affecting review quality specifically, the difficulty in comprehending code stands out as one of the most commonly cited. A survey conducted by Kononenko et al. (2016) shows that developers consider understanding the to be evaluated code a major challenge. Similarly, Bacchelli & Bird (2013) have shown that understanding code is an integral aspect of code review as it allows reviewers to analyse the code more quickly and provide more valuable feedback. Unfortunately, qualitative as well as quantitative research shows that understanding code can be a time-consuming and mentally demanding task, particularly when dealing with complex code (Huang et al., 2012; Peitek et al., 2021).

2.3 Code complexity and complexity metrics

Code complexity is a common concept in software engineering and describes the level of difficulty involved in comprehending, modifying, and maintaining source code (Curtis et al., 1979). It is a multidimensional construct that, depending on the exact metric, encompasses factors such as code size, path complexity, and various others (Zuse, 1991). Consequently, numerous metrics have been developed to quantify, measure and track complexity in practice. These different complexity metrics quantitatively represent different aspects of the software (Nuñez-Varela et al., 2017).

One of the most widely known complexity metrics is McCabe's cyclomatic complexity (V(g)), which assesses the number of independent paths that can be taken through a piece of code (McCabe, 1976). A higher cyclomatic complexity value indicates a more complex codebase with a greater number of decision points and possible paths. Although popular, the metric inadequately captures complexity arising from nesting, wherein a control structure is enclosed within another control structure (Shepperd, 1988; Sarwar et al., 2013). This means that code with equal cyclomatic complexity can have very different levels of nesting depth. Consequently, nesting depth has become another important measure and represents the number of control structures (e.g., if statements, loops) that are nested within one another. A higher nesting depth value indicates a more complex, deeply nested code. (Alrasheed & Melton, 2022; Harrison & Magel, 1981).

2.3.1 Code complexity and understandability

Subjective perceptions of developers suggest that increasingly complex code becomes increasingly difficult to understand and consequently review. For example, Kononenko et al. (2016) interviewed developers and found that they consider larger and more complex code difficult and time-consuming to review, as it makes understanding the code in general as well as the code change in particular a more demanding cognitive task. Likewise, Ebert et al. (2021) found that the most frequent reason for confusion during code review is the length or complexity of a code change. Comparing 11 complexity metrics, a survey by Antinyan et al. (2017) revealed that most software engineers consider lack of structure and nesting depth to have the biggest influence on complexity and subsequent understandability.

A recent quantitative study by Peitek et al. (2021) supports these findings. Making a first attempt to investigate the relationship between code complexity metrics, comprehension, and its neural correlates, they showed that more complex code reduces comprehension and puts increased demand on brain areas related to cognitive effort. Participants were asked to read and understand Java snippets with varying levels of various complexity metrics while their brain

activity was recorded using functional magnetic resonance imaging (fMRI). The study found a negative correlation between code complexity metrics such as cyclomatic complexity and comprehension accuracy, indicating that more complex code is harder to understand.

2.3.2 Code complexity and mental workload

In addition to showing that complexity reduces understandability of code, Peitek et al. (2021) showed that brain regions activated during program comprehension were associated with cognitive control, working memory, and attention. This indicates that cognitive load may play a central role in code understanding and consequently reviewer performance. It refers to the amount of mental effort, working memory capacity, and attentional resources that is required to perform a task successfully (Sweller, 1988) and is influenced by various factors such as task complexity, information processing demands, and individual cognitive abilities (Sweller et al., 2011a). For example, in the context of code review, reduced mental load and increased working memory capacity have been shown to increase review performance (Baum et al., 2017, 2019).

While cognitive load refers to the objective amount of mental effort and resources required for a task, mental workload refers to the subjective perception of that effort and can be assessed using various scales, such as the NASA Task Load Index (NASA-TLX; de Winter, 2014; Sweller et al., 2011b). While there is a substantial amount of research around complexity, metrics, and their role in software quality, to date, only few studies have focused on their influence on human mental workload, particularly in code review. To support this claim, a search for relevant literature was conducted on the metadata, specifically on the title, abstract and keywords. Given that human factors engineering is a multidisciplinary field, and this research lies at the junction of software engineering and psychology, multiple electronic databases were chosen to search for relevant literature. Web of Science and Scopus databases were selected as they are multidisciplinary and likely to yield diverse studies. Additionally, IEEE was included as it hosts journal papers related to the field of engineering specifically. The criteria for inclusion were as follows:

- An article should have quantitative empirical work (i.e., involve the use of actual quantitative data) illustrating the link between at least one code complexity measure and any of the dependent variables under investigation, namely workload, review efficiency, or effectiveness.
- An article should relate to human, not automated performance.

The exclusion of papers was done by screening the titles and abstracts against the inclusion criteria. The remaining relevant papers were then assessed based on their full text. Papers that did not fulfil all three of the aforementioned criteria were excluded.

Table 1

Search terms used for systematic review and respective hits after exclusion.

Search Term	IEEE Hits	Web of Science Hits	Scopus Hits
("code review" OR "software review" OR "reviewing code") AND "complexity" AND "workload"	1	0	1
("code review" OR "software review" OR "reviewing code") AND "cyclomatic complexity" AND "workload"	0	0	0
("code review" OR "software review" OR "reviewing code") AND "nesting depth" AND "workload"	0	0	0

Excessive workload has been consistently linked to performance losses across contexts (e.g. Bruggen, 2015; Dehais et al., 2020; Fan & Smith, 2017). Thus, when code reviewers experience overwhelming levels of complexity, it may lead to overall reduced review performance. Therefore, to ensure sustainable workload levels during code review and optimise the effectiveness of the review process, it is crucial to first gain a comprehensive understanding of how code complexity impacts reviewers' workload. In the context of code review, only one study was found to have investigated mental workload in relation to complexity metrics (Table 1). In this study, Hijazi et al. (2023) compared the mental effort ratings on the NASA-TLX scale between more complex programs and less complex programs during code review. They found that the more complex programs had comparatively higher mental effort ratings on the NASA-TLX. However, the study did not examine the precise impact of increasing complexity metrics on these outcomes. Consequently, we aim to answer the following research questions:

$\boldsymbol{RQ_1}$: What is the impact of cyclomatic complexity on subjective mental workload during code review?

$\boldsymbol{RQ_2}$: What is the impact of nesting depth on subjective mental workload during code review?

2.3.3 Code complexity and review performance

While above research suggests that code complexity may have a significant impact on code comprehension and workload, little is known about the impact of the different complexity

metrics on reviewer performance. Review performance in the context of code review refers to the ability of reviewers to work efficiently and effectively. In human factors research, the former describes the amount of time it takes an individual to complete a task while the latter describes the ability of an individual to achieve a desired outcome (Wickens, 2014). In the context of code review, they therefore refer to the time needed to finish a review and the ability of a reviewer to detect shortcomings respectively.

Previous research provides initial evidence that code complexity may negatively affect review performance. Subjective experiences of software developers suggest that complexity may negatively affect review time. When asked to estimate the additional time needed to maintain complex code compared to simple code, most software engineers believe complex code to increase maintenance time by a factor of at least two (Antinyan et al., 2017). Furthermore, Baum et al. (2019) examined the relationship between code changes and defect detection effectiveness and found that larger, more complex code changes are associated with lower defect detection effectiveness for delocalised defects. This suggests that complexity may in fact have a substantial effect on performance outcomes of individual code reviewers. Yet, most research does not measure the impact of complexity metrics specifically. To date, no studies have investigated the impact of increasing code complexity metrics on code review performance in a controlled experimental setting (Table 2).

Table 2

Search terms used for systematic review and respective hits after exclusion.

Search Term	IEEE Hits	Web of Science Hits	Scopus Hits
(“code review” OR “software review” OR “reviewing code”) AND (“complexity” OR “cyclomatic complexity” OR “nesting depth”) AND (“fault detection” OR “bug detection” OR “detecting faults” OR “detecting bugs” OR “debug”)	1	1	1
(“code review” OR “software review” OR “reviewing code”) AND (“complexity” OR “cyclomatic complexity” OR “nesting depth”) AND (“defect” or “bug”)	1	1	1
(“code review” OR “software review” OR “reviewing code”) AND (“complexity” OR “cyclomatic complexity” OR “nesting depth”) AND “performance”	0	0	0
(“code review” OR “software review” OR “reviewing code”) AND (“complexity” OR “cyclomatic complexity” OR “nesting depth”) AND “effectiveness”	0	0	0

("code review" OR "software review" OR "reviewing code") AND ("complexity" OR "cyclomatic complexity" OR "nesting depth") AND ("efficiency" OR "time")	1	1	1

In their recent paper, Hijazi et al. (2023) used a combination of experience level, cognitive load as measured by heart rate variability and task-evoked pupillary responses, time on task, and code complexity measures to predict review quality in terms of defect detection effectiveness. While they found that this approach can predict review quality with roughly 87 percent accuracy, they did not systematically investigate the impact of increasing code complexity on review performance. Therefore, this study aims to fill this gap. By using a within-subject design and systematically varying code complexity metrics it aims to provide quantitative insights into the impact of code complexity on code review performance. Specifically, it investigates the effects of cyclomatic complexity and nesting depth on review efficiency (i.e., time on task) and effectiveness (i.e., defect detection) during code review. Accordingly, we aim to answer the following research questions:

$\boldsymbol{RQ_3}$: What is the impact of cyclomatic complexity on time on task during code review?
$\boldsymbol{RQ_4}$: What is the impact of nesting depth on time on task during code review?
$\boldsymbol{RQ_5}$: What is the impact of cyclomatic complexity on defect detection during code review?
$\boldsymbol{RQ_6}$: What is the impact of nesting depth on defect detection during code review?

Both efficiency and effectiveness are central to a successful code review process. Improving efficiency allows for faster development cycles while enhancing effectiveness improves the overall quality of the code. By considering the impact of different complexity metrics on reviewer performance, insights into how different kinds of complexity influence both the efficiency and effectiveness of code review can be gained. This knowledge can inform best practices for code review and inspire the development of strategies, tools, and guidelines that optimise the review process.

3 Methods

3.1 Participants

17 participants were recruited from the embedded software development division of Vector Informatik GmbH, a large software company. All participants were experienced embedded C programmers. The experiment was approved by the faculty's ethics committee and all participants provided informed consent before taking part in the study.

3.2 Design

The study used a within-subjects design, in which each participant reviewed eight code snippets with varying levels of cyclomatic complexity and nesting depth. The independent variables were the level of cyclomatic complexity and nesting depth. The dependent variables were subjective workload, defect detection, and time on task. For each code snippet, subjective workload was measured using the NASA-TLX scale, defect detection was measured as a binary (yes/no) variable and time on task was measured as minutes spent on reviewing a snippet.

3.3 Materials

3.3.1 Microsoft Forms

Microsoft Forms was used as the survey tool to collect data from participants. It is not only easily accessible and user friendly. It also provides all the necessary functionality and is the commonly used survey tool that participants are already familiar with, thereby enhancing ease of use.

3.3.2 Code snippets

The study used eight code snippets, each categorised by three levels of cyclomatic complexity and three levels of nesting depth. The cyclomatic complexity levels were low (CC = 0-9), medium (CC = 10-19), and high (CC = 20-30). The nesting depth levels were low (ND = 0-2), medium (ND = 3-5), and high (ND = 6-8). To ensure that the code snippets used in the study were representative of real-world code, existing snippets utilised at Vector were used as a basis. Each snippet was written in C and entailed exactly one defect. If necessary, the defect was artificially produced. Subsequently, three experienced software developers reviewed the snippets to confirm their representativeness. Table 3 summarises the characteristics of each code snippet.

Table 3

Code snippets used in the study and their respective complexity metric scores.

Snippet	Cyclomatic Complexity	Nesting Depth	LOC
1	Low (CC = 3)	Low (ND = 1)	49
2	Low (CC = 6)	Medium (ND = 4)	96
3	Low (CC = 9)	High (ND = 7)	125
4	Medium (CC = 12)	Low (ND = 2)	117
5	Medium (CC = 15)	Medium (ND = 5)	222
6	Medium (CC = 18)	High (ND = 8)	273
7	High (CC = 21)	Low (ND = 3)	183
8	High (CC = 27)	High (ND = 6)	370

3.3.3 Review Tool

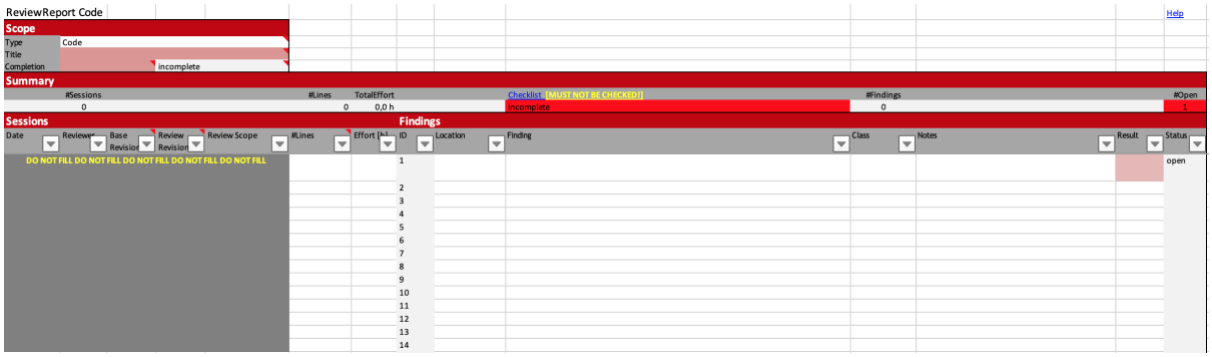
To ensure that code review is performed in a manner that reflects usual practices and preferences, participants were given the flexibility to use the review tool of their choice during. Commonly used review tools include text editors like Microsoft Visual Studio, Visual Studio Code, Eclipse CDT, vim, Notepad ++, or any other text editor that supports syntax highlighting.

3.3.4 Review report form

The review report form used to collect data on review outcomes and time needed was an Excel sheet, which is a commonly utilised tool for code review in the participants' organisation. The Excel sheet was specifically designed to capture only relevant information during the code review process, namely details on identified defects, and the time spent on each review. Participants simply had to report the line in which they found a defect, the nature of that defect, what needs to be changed, and how long the review took. Any fields that contained personally identifiable information or data that was not relevant to the research objectives were omitted (Figure 1). By utilising the existing review report form, we aimed to ensure familiarity and ease of use for the participants, minimising any additional cognitive load that may arise from adapting to an unfamiliar tool.

Figure 1

Review report form.



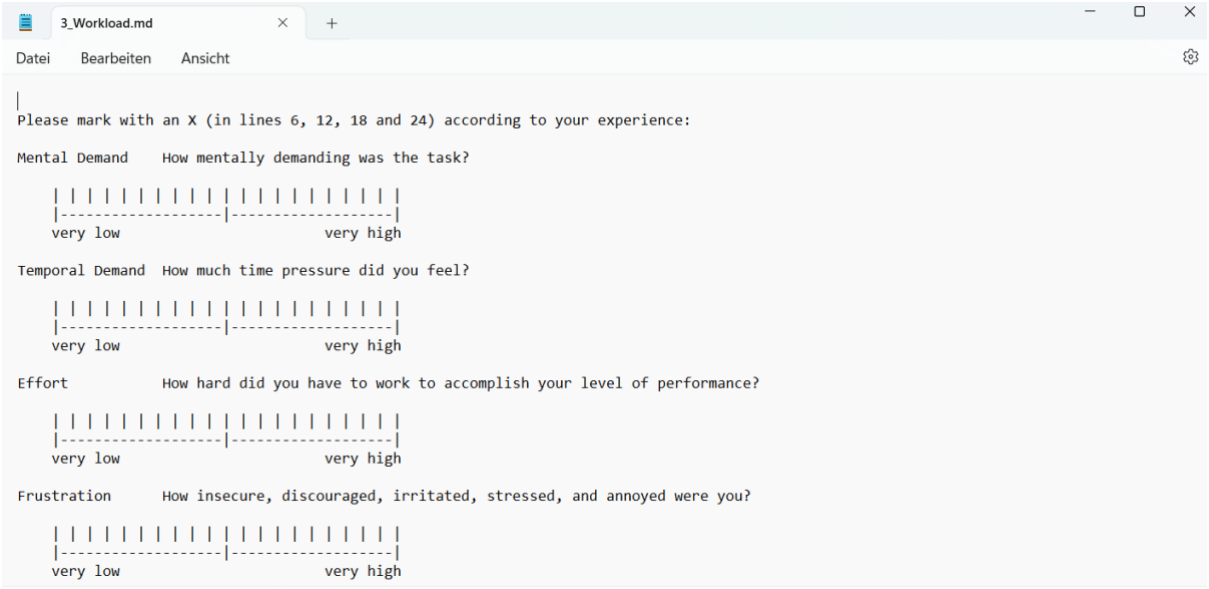
3.3.5 NASA-TLX

The NASA-TLX is used to assess and quantify the perceived workload during code review, the third dependent variable. It is a subjective workload questionnaire that is widely used and validated in human factors research (de Winter, 2014; Sweller et al., 2011b). In general, the overall workload rating is based on the average of six dimensions, including (1) mental demand, (2) physical demand, (3) temporal demand, (4) performance, (5) effort, and (6) frustration (Hart, 2006; Hart & Staveland, 1988). The physical demand dimension was deemed irrelevant in the context of code review. Likewise, the performance dimension was deemed irrelevant as performance was assessed by objectives measures, i.e., time on task and fault detection. Therefore, these dimensions will be excluded from the scale employed for the present study.

Consequently, this study focused on four remaining key dimensions of mental workload: mental demand, temporal demand, effort, and frustration (Figure 2). The mental demand scale measured the mental activity required to perform the review and provided insights into the complexity of cognitive processes such as code understanding and defect detection. The temporal demand scale provided insights into the time pressure experienced by participants during review. The effort scale assessed the perceived level of mental effort participants had to recruit to accomplish the respective level of performance and highlighted the intensity of cognitive engagement required for the review. Lastly, the frustration scale measured participants' negative emotional reaction, namely their level of annoyance, stress, and irritation experienced during the review. As we were interested in each measure separately, the workload ratings were not averaged across scales.

Figure 2

Modified NASA-TLX.



3.4 Procedure

Each participant completed a pre-test survey (Microsoft Forms) to collect demographic information, self-reported measures of programming experience, familiarity with the C programming language, frequency of programming and code review, tiredness (i.e., working hours before the experiment), perceived fitness, and confidence in the ability to detect mistakes. After completing the survey, participants received a compressed file with the review materials. It included eight distinct folders, each containing one snippet, one report form and one NASA-TLX form. Participants were given unlimited time to review each code. After downloading the file, participants were required to (1) use their preferred tool to review the code snippet, (2) report any detected defects and the self-measured duration of the review in the report form, and (3) complete the NASA-TLX. As there were eight snippets, participants had to perform these three steps eight times, one time per snippet. They were then requested to upload the compressed file via the provided forms. As participants were performing the code review according to their familiar procedures and using familiar tools, a practice session was omitted.

3.5 Data Analysis

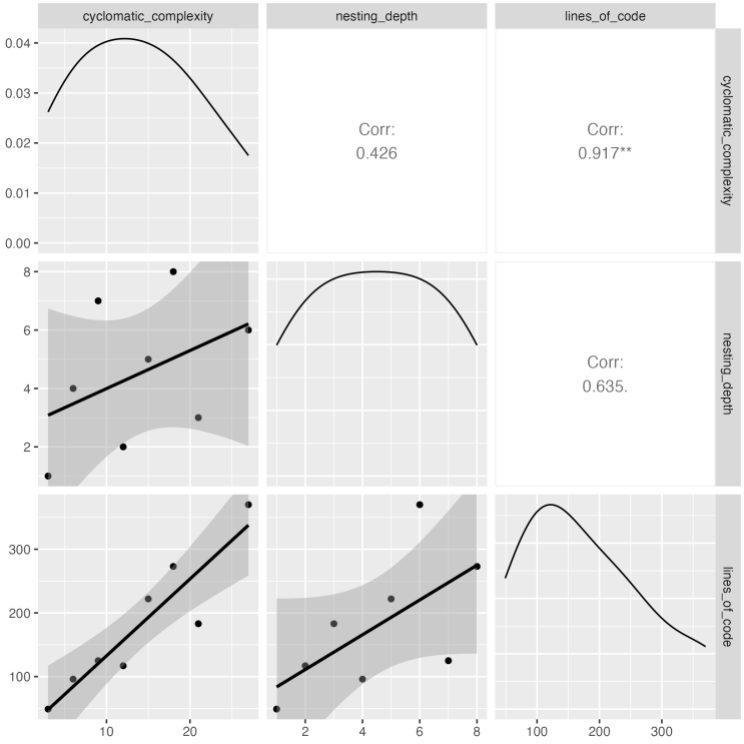
The data collected in this study were analysed using R statistical software (version 2022.07.2). The analysis aimed to explore the relationships between the independent variables, namely cyclomatic complexity and nesting depth, and the dependent variables, namely workload, time on task and defect detection. First, the data were imported and cleaned. The survey results were read from the "MS_results.csv" file, and the experimental results, were read from the "Snippet_results.csv" file. Next, the data was joined to create a merged dataset

that included both the survey results and the experimental results. The variables were appropriately renamed and converted. Descriptive analyses by means of scatterplots and line plots were conducted to summarise the data and gain a preliminary understanding of the relationship between variables.

In a first attempt to analyse the data, the exact model specification used for the analysis was determined using the glmulti package, a program used to explore multiple potential model configurations (Calcagno & de Mazancourt, 2010). However, these models showed paradoxical results and were not interpretable, likely due to the small sample size. In addition, the variable lines of code (LOC) was initially considered as a potential control variable in the analysis. However, prior to inclusion in the regression models, the correlation between LOC and the complexity metrics was investigated, to check for possible multicollinearity. As LOC exhibited high correlations with both complexity metrics (Figure 3) and regression analyses including lines of code as a predictor yielded paradoxical results, with unexpected and contradictory relationships between the variables, we made the decision to exclude LOC from the final analysis to mitigate the potential issue of multicollinearity and ensure more reliable and interpretable results.

Figure 3

Correlations between predictor variables.



Therefore, we chose to explore only the population data of the theoretically minimal model and the reported findings and interpretations focus solely on the effects of cyclomatic complexity and nesting depth on the outcome variables of interest. To examine this relationship, a multivariate regression model was fitted using the brm function from the brms package (Bürkner, 2017). Fixed effect estimates were extracted using the summary function.

4 Results

4.1 Complexity metrics and workload

To examine the impact of these complexity metrics on perceived workload, a multivariate regression model including all workload scales was fitted. A graphical presentation of the correlations and multi-level effects of cyclomatic complexity and nesting depth on workload ratings can be found in figure 4 and 5 respectively. Coefficient estimates, along with their 95% credibility intervals, are shown in Tables 4-7.

Figure 4

Correlations between complexity metrics and workload.

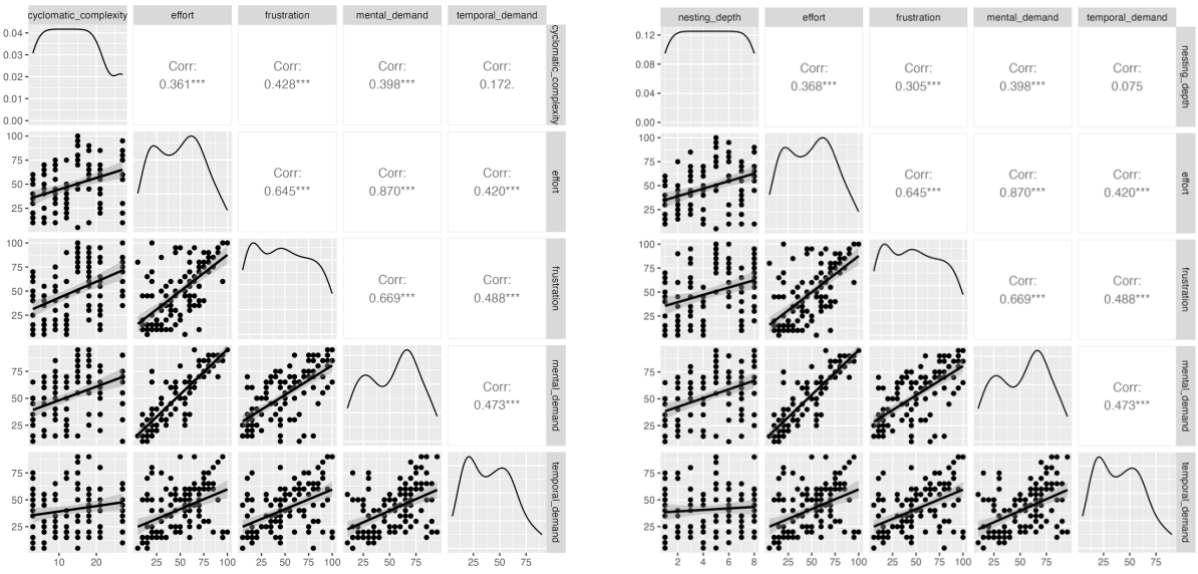
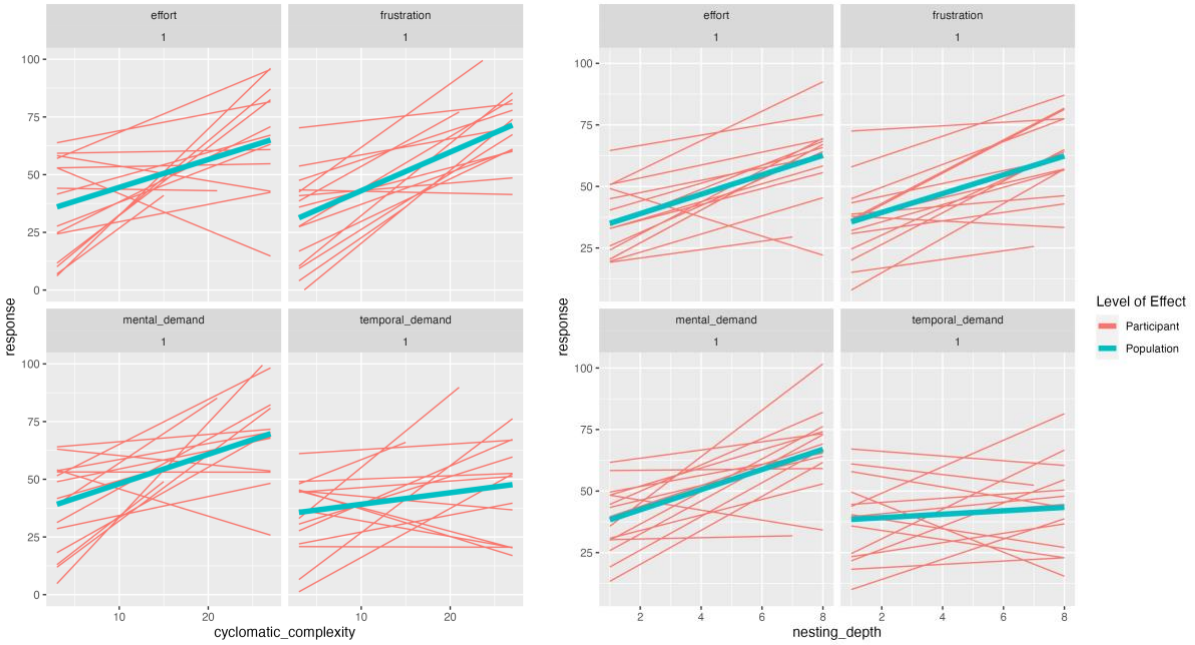


Figure 5

Multilevel plots of the effect of cyclomatic metrics on workload.



4.1.1 Effort

Figure 4 shows moderate positive correlations between cyclomatic complexity and effort (r = .361) and between nesting depth and effort (r = .368). However, the multilevel plots in Figure 5 reveal that the effect of cyclomatic complexity on effort varies among individual participants. This suggests that the observed positive association may not be universally applicable. One participant even shows the inverse relationship. Yet, this participant can be considered an outlier as they abandoned the review of the more complex code snippets, subsequently reporting lower workload ratings. On the other hand, the relationship between nesting depth and effort consistently shows a positive trend among the majority of reviewers, with only the outlier displaying a negative trend.

The regression model supports these observations (Table 4). Specifically, for every unit increase in cyclomatic complexity, there is an associated increase in effort of an estimated 0.76 points. As cyclomatic complexity spans from zero to 27, this amounts to an average total increase of roughly 20 points. Similarly, for nesting depth, a one unit increase in nesting depth increases effort by an estimated 2.79 points. As nesting depth spans from zero to eight, the model suggests an estimated total effort increase of about 22 points. The credibility intervals for both complexity metrics do not include zero, therefore the true association is likely to be positive. However, both intervals are wide, making the exact magnitude of the impact of complexity on effort uncertain.

Table 4

Regression coefficients for effort.

Predictor	Centre	Lower	Upper
Intercept	25.75	15.61	35.76
Cyclomatic Complexity	0.76	0.12	1.37
Nesting Depth	2.79	0.72	4.76

4.1.2 Frustration

The data shows a positive population effect of both complexity metrics on frustration, with a slightly stronger association observed for cyclomatic complexity (Figure 4). Specifically, the correlation between cyclomatic complexity and frustration is .428, while that between nesting depth and effort is .305. The positive trend is observed in all participants except for one identified outlier (Figure 5).

Consistent with the correlational findings, the multivariate regression model confirms the positive effects of both complexity metrics on frustration (Table 5). Each unit increase in cyclomatic complexity leads to an average increase in frustration of 1.45 points. This means that the highest level of cyclomatic complexity increases frustration by almost 40 points on average. The credibility interval (95% CI: [0.73, 2.18]) provides confidence that the true effect is positive. The effect of nesting depth is slightly smaller, with each unit increase resulting in an average increase in frustration of 1.82 points. Therefore, the highest level of nesting depth increases frustration by an average of 15 points. However, the credibility interval for the nesting depth includes zero (95% CI: [-0.51, 4.26]), resulting in a higher degree of uncertainty in the true effect of nesting depth on frustration.

Table 5

Regression coefficients for frustration.

Predictor	Centre	Lower	Upper
Intercept	21.26	9.54	33.17
Cyclomatic Complexity	1.45	0.73	2.18
Nesting Depth	1.82	-0.51	4.26

4.1.3 Mental demand

Overall, the participant data seems very similar to the effort scale. There seems to be a positive population effect of both complexity metrics on mental demand, with both correlations being .398 (Figure 4). However, when examining the individual participant data, some variability is evident, particularly for cyclomatic complexity, suggesting that the positive trend may not be universally applicable (Figure 5). Even after exclusion of the outlier two participants do not show a positive association. The picture is clearer when looking at nesting depth. Excluding the outlier, all participants show a positive trend.

The multivariate regression model confirms the positive effects of both complexity metrics on mental demand (Table 6). Cyclomatic complexity showed an estimated coefficient of 0.84, meaning that for each additional unit of cyclomatic complexity, mental demand increased only by approximately 0.84 points. Thus, on average, the highest level of cyclomatic complexity increases frustration by almost 23 points. Nesting depth appeared to have the same effect on mental demand, with an estimated coefficient of 2.86. Like cyclomatic complexity, the highest level of nesting depth increases frustration by almost 23 points on average. Again, it is important to note the credibility intervals surrounding these estimates (95% CI: [0.26, 1.43] and [0.97, 4.65] respectively), indicating some uncertainty about the exact magnitude of these effects.

Table 6

Regression coefficients for mental demand.

Predictor	Centre	Lower	Upper
Intercept	28.19	18.94	38.12
Cyclomatic Complexity	0.84	0.26	1.43
Nesting Depth	2.86	0.97	4.65

4.1.4 Temporal demand

The data reveals weak correlations between complexity metrics and temporal demand, with cyclomatic complexity showing a correlation of 0.172 and nesting depth showing a correlation of 0.075 (Figure 4). The multilevel plots also indicate no clear effect of complexity metrics on temporal demand, as many participants' temporal demand remained stable across complexity levels (Figure 5). Some participants experienced a slight increase in temporal demand with higher complexity, while others experienced the opposite. These findings suggest that any impact of complexity on temporal demand is not consistent among individuals.

Consistent with the correlational data, the multivariate regression model shows that both cyclomatic complexity and nesting depth show only small associations with temporal demand (Table 7). Cyclomatic complexity demonstrates an estimated coefficient of 0.55, suggesting that for each unit increase in cyclomatic complexity, the temporal demand increases by an average of 0.55 points. Thus, the highest level of cyclomatic complexity leads to a temporal demand that is approximately 15 points higher than the lowest level. Nesting depth has a smaller positive association, with each unit increase resulting in a 0.02-point increase in temporal demand. At the highest nesting level, the increase in temporal demand is only 0.16 points. However, it is important to note that the credibility intervals include zero, indicating a high level of uncertainty in the effect of these metrics on temporal demand. Overall, the relationship between complexity metrics and temporal demand appears to be limited, considering the wide confidence intervals and relatively small coefficients.

Table 7

Regression coefficients for temporal demand.

Predictor	Centre	Lower	Upper
Intercept	33.64	24.01	43.23
Cyclomatic Complexity	0.55	-0.05	1.16
Nesting Depth	0.02	-1.95	1.90

4.2 Complexity metrics and performance

4.2.1 Time on task

Both complexity metrics show a positive association with time on task (Figure 6). While cyclomatic complexity has a correlation coefficient of .292, nesting depth exhibits a stronger correlation coefficient of .426. The multilevel plots further illustrate the nature of these effects (Figure 7). The relationship between nesting depth and time on task appears to be universal, with all participants displaying a positive trend. On the other hand, the effect of cyclomatic complexity on time on task varies significantly among individual participants. Some participants show little to no impact of cyclomatic complexity on time on task, while others exhibit a reverse relationship, i.e., increased complexity corresponds to decreased time on task.

Qualitative analysis of the review forms revealed that several participants decided to skip the review of highly complex code snippets, specifically snippets 5, 6, and 7. Consequently, these participants reported a notably reduced time on task (as low as 5 minutes) or did not

record any time at all. This behaviour has likely distorted the true effect of complexity on the time needed to review the code Regarding nesting depth, there is a consistent positive association between complexity and time on task. Higher nesting depth corresponds to a greater amount of time needed for code review. This effect appears to be universal, as the majority of participants exhibit a positive trend in their slopes.

Figure 6

Correlation plot of the effect of cyclomatic complexity and nesting depth on ToT.

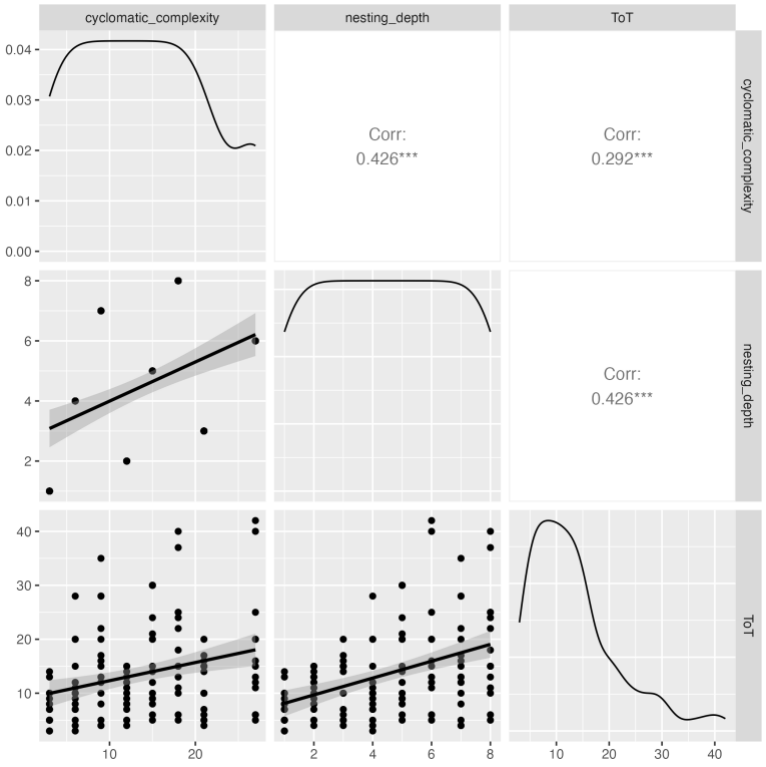
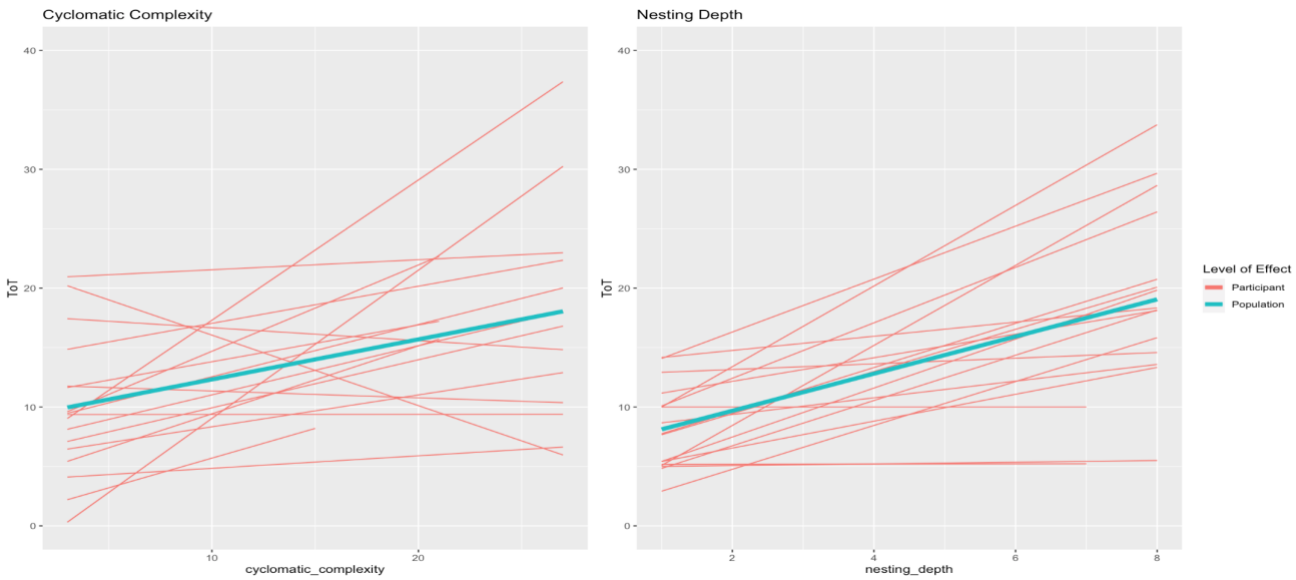


Figure 7

Multilevel plot of the effect of cyclomatic complexity and nesting depth on ToT.



Overall, the fitted regression model supports this interpretation (Table 8). A one-unit increase in nesting depth is associated with an average increase in review time of approximately 1.37 minutes. This amounts to a total average increase of 11 minutes. While there is some uncertainty surrounding the precise magnitude of the effect, as indicated by the moderately wide credibility interval (95% CI: [0.69, 2.06]), we can confidently say that this effect truly exists. In contrast, the relationship between cyclomatic complexity and time on task appears to be less pronounced. The estimated coefficient for cyclomatic complexity is 0.14, indicating that a one-unit increase in cyclomatic complexity leads to an average review time increase of approximately 0.14 minutes. In other words, at the highest level of cyclomatic complexity, reviewers require around 4 minutes more to complete the code review. However, this effect needs to be considered with caution given that the credibility interval (95% CI: [-0.08, 0.35]) includes negative values as well as zero as possible outcomes.

Table 8

Regression coefficient estimates for time on task.

Predictor	Centre	Lower	Upper
Intercept	5.30	1.76	8.89
Cyclomatic Complexity	0.14	-0.08	0.35
Nesting Depth	1.37	0.69	2.06

4.2.2 Defect detection

Figure 8 shows that there is a negative correlation between both complexity metrics and fault detection. Specifically, cyclomatic complexity shows a correlation coefficient of -.261, while nesting depth shows a slightly smaller negative correlation of -.193.

Figure 8

Correlation plot of the effect of cyclomatic complexity and nesting depth on defect detection.

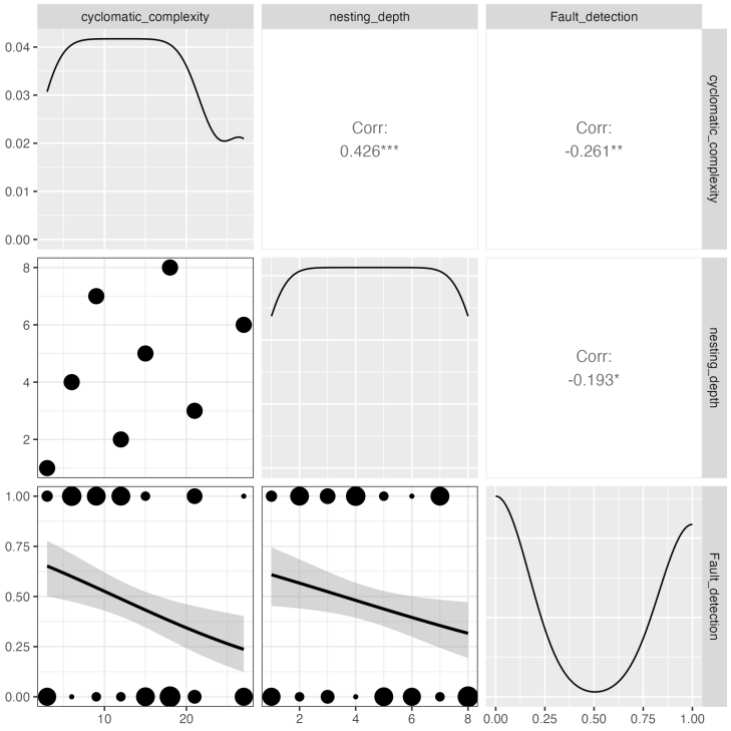


Table 9 presents the estimates, along with their corresponding confidence intervals, for the population-level effects of the complexity metrics on defect detection. The results indicate that higher values of both cyclomatic complexity and nesting depth are associated with a reduced likelihood of detecting defects during code review. Specifically, cyclomatic complexity shows a negative effect on defect detection (odds = 0.94), implying that for each unit increase in cyclomatic complexity, the odds of detecting defects decrease by approximately 6%. The narrow credibility interval (95% CI: [0.88, 0.99]) provides reasonable confidence in the existence of this effect. Similarly, nesting depth was found to have a negative effect on defect detection (odds = 0.92), with each unit increase in nesting depth decreasing the odds of detecting defects by approximately 8%. However, unlike for cyclomatic complexity, the credibility interval for nesting depth is wide (95% CI: [0.76, 1.07]) and includes 1 as a possible outcome, making the effect of nesting depth uncertain.

Table 9

Regression coefficient estimates for the odds of defect detection.

Predictor	Centre	Lower	Upper
Intercept	2.71	1.05	7.10
Cyclomatic Complexity	0.94	0.88	0.99
Nesting Depth	0.92	0.76	1.07

5 Discussion

The aim of the present study was to examine the impact of code complexity, i.e. cyclomatic complexity and nesting depth on various aspects of code review performance, i.e., mental workload, review efficiency, and review effectiveness. Initially, it was intended to include LOC as a control variable in the analysis. However, due to strong correlations with the other metrics, problems stemming from multicollinearity led to distortions in the estimation of their individual effects. Furthermore, the decision to omit LOC is supported by the broader literature on code quality. Using LOC as a measure of performance and quality outcomes has long been subject to criticism. Numerous studies have challenged the validity and reliability of using LOC as an indicator of code quality and performance (e.g. Barb et al., 2014) and metrics such as nesting depth have emerged as more robust indicators of code vulnerabilities (Shin & Williams, 2008).

5.1 Complexity metrics and workload

Cognitive load theory predicts that more complex tasks should increase the mental work needed to perform a task (Sweller et al., 2011a). Consistent with this theory, previous research has shown first indications that more complex code leads to higher subjective workload ratings (Hijazi et al., 2023), and that this effect is reflected in brain activity (Peitek et al., 2021). Our study supports and extends these findings, demonstrating the magnitude with which complexity metrics, such as cyclomatic complexity and nesting depth, increase the perceived mental and effort required for code review. Specifically, at their maximum, these metrics similarly increase mental demand and effort by a little more than 20 points on average. This aligns with previous reports of developers who noted that more complex code is considered more difficult to review (Kononenko et al., 2016).

Interestingly, our study revealed that despite showing similar impact across their range, that different complexity metrics differ in the strength with which they affect mental demand and effort. Specifically, a one-step increase in nesting depth shows a stronger impact on mental demand and effort than cyclomatic complexity. A potential explanation may be that nesting depth increases mental demand and effort in a less linear manner than cyclomatic complexity. For cyclomatic complexity, each additional path introduces an additional, yet distinct piece of logic to keep track of. Thus, the load imposed on cognition increases in a linear manner. However, for nesting depth, each nested level introduces a new context. Consequently, the code reviewer must understand the logic at the current level, while still retaining the contexts of all enclosing levels. Switching between these contexts and understanding their interplay is a more cognitively challenging task, significantly increasing the need for logical reasoning (Alrasheed & Melton, 2022). As such reasoning requires one to retain several mental models in working memory, each level of nesting significantly increases working memory demand (Barrouillet & Lecas, 1999). To examine this differential effect on working memory, future research could introduce additional tasks that put load on working memory capacity while participants engage in code review. This would allow for a more comprehensive understanding of how different complexity metrics influence such cognitive processes during code review.

Despite having unequal impact on mental demand and effort, a one-step increase in cyclomatic complexity and nesting depth on average increase frustration in a similar manner. Consequently, as cyclomatic complexity ranges until 27, it increases frustration by an average of 40 points at its maximum. As nesting depth only ranges until 8, its highest level merely increases frustration by an average of 15 points. Frustration, as a psychological construct, is often associated with the experience of encountering obstacles while trying to achieve a goal and represents a more general emotional response to difficulty (Berkowitz, 1989). The similar influence on frustration on a one-step level, despite the differential impact on mental demand and effort, supports this notion. It suggests that frustration in code review may be primarily driven by the general increase in obstacles, that is added complexity levels, regardless of the specific form these obstacles take.

Finally, on average, temporal demand did not seem to be affected much by code complexity. However, the effect of temporal demand varied substantially between people. For many participants, complexity did not show any influence on temporal demand and there are several explanations for this. First, as the experiment was not time constraint, it did not impose any inherent temporal pressure, perhaps introducing validity issues. In real life, reviewers have reported to experience time pressure – formally or informally – and our results may not be

representative (Kononenko et al., 2016). Future research should consider incorporating time constraints that align with company practices and pressures. Some companies may prioritise thoroughness and encourage reviewers to invest more time in their assessments, while others may emphasise speed and quick turnaround. Second, reviewers reported experiencing high levels of time pressure even in the absence of code complexity. Conducting the study during regular working hours may have influenced participants' subjective experience of time pressure, as they were likely juggling their regular work responsibilities alongside the experimental task. This pre-existing time pressure could have overshadowed any additional temporal demand introduced by manipulating code complexity. Future studies could consider conducting experiments during dedicated time slots or adjusting the workload distribution to minimise external time pressures. Third, there is a possible self-selection is bias. Developers that currently experience, or frequently suffer from time pressure, were less likely to sign up to the study. As a result, the sample of reviewers in the experiment may have consisted of individuals who are less affected by time pressure.

However, some participants did in fact observe either a positive or negative association between complexity and time pressure. Such individual differences may have stemmed from perceived relevance and consequent task engagement. Participants who are engaged and motivated may invest more time and effort into thoroughly reviewing complex code, resulting in a positive association between complexity and temporal demand. Conversely, participants who are less motivated may not allocate as much time to complex code, leading to a weaker or non-existent association. For those reviewers showing a negative association, it is possible that they counteracted increasing pressure by switching strategies to only skim the code as it got more complex. This may have offset subjective temporal demand with increasing complexity. Future research could leverage technologies such as eye-tracking to explore whether increased code complexity triggers changes in code review strategies.

5.2 Complexity metrics and performance

5.2.1 Review efficiency

This study highlights the fact, that increasing code complexity can in fact affect the efficiency of code review. It was found that higher levels of nesting depth were associated with increased time on task, suggesting that reviewers must spend more time understanding, navigating, and evaluating the code as its structure becomes more nested. Specifically, for each added nesting level, the review time on average increases by a little more than a minute. This implies that the inherent complexity associated with deeper nesting presents a cognitive

challenge for reviewers, affecting their ability to efficiently review code. This was expected as the review of many execution paths was likely more challenging comprehend, which in turn increased the time required to complete the code review. Contrastingly, the study found that cyclomatic complexity has a much smaller effect on time on task, increasing review time only by a few seconds per level. Thus, at the highest complexity level, cyclomatic complexity increases review time by only 4 minutes, while nesting depth increases it by 11 minutes on average. This aligns with previous reasoning that cyclomatic complexity may be easier to manage than complexity arising from nesting depth. While this increase in time may seem small, over the course of a project these additional seconds or minutes can aggregate into a substantial amount of time, potentially leading to significant delays.

Interestingly, this study further found that reviewers might abandon the review when faced with excessive complexity. This finding raises questions about the psychological effects of complexity on reviewer motivation and perseverance. For example, a study by Wang et al. (2008) found that reviewers are not willing to do their best work when they are assigned to poorly written code. It is possible, that code complexity that is considered “too high” may have a similar effect. Future research should investigate the existence of a complexity "tipping point" beyond which reviewers are more likely to give up on the review process or reject a pull request from the onset. With code review being the last line of defence against defects, such behaviour has the potential to result in serious defects and severe consequences like in the case of Toyota. However, it is important to note that the participants in this study were aware that the code review was conducted solely for experimental purposes. Such awareness may influence perceptions of the potential consequences of errors, likely lowering the threshold for giving up. In real-world code review scenarios, where the stakes are perceived as higher, this threshold may be much higher as well.

Such motivational factors may further provide a possible explanation for the observed between participant variability. Multilevel plots (Figure 7) indicate that the effect of complexity on the time spent to review the code may vary between people. Such individual differences may stem from various factors such as expertise (Sharif et al., 2012) and working memory capacity (Baum et al., 2019) but perhaps also differences in intrinsic motivation. Individuals with strong intrinsic motivation for code review may exhibit higher levels of persistence, effort, and focus when faced with complex code, leading to more effective and efficient review processes. On the other hand, individuals with lower levels of intrinsic motivation may be more prone to experiencing frustration and reduced performance when encountering increasing complexity (Chen & Caza, 2018). As defect detection has been linked to the rigour with which

code is reviewed, such behavioural reactions to complexity can have detrimental consequences for code quality (Thongtanunam et al., 2015).

5.2.2 Review effectiveness

Overall, this study demonstrates that higher levels of both cyclomatic complexity and nesting depth are associated with a decreased likelihood of detecting defects during code review. This aligns with previous research by Baum et al. (2019) who were the first to find that larger, more complex code change seems to reduce defect detection effectiveness during review, at least for delocalised defects. Our study provides a more detailed understanding of these findings by providing insights into the exact magnitude with which specific complexity metrics reduce this effectiveness. Specifically, we found that each additional level of cyclomatic complexity reduces the odds of detecting defects by 6.1% while each additional level of nesting reduces these odds by 8.9%.

Considering that we could confirm previous studies demonstrating that complexity also increases mental workload (Hijazi et al., 2023; Peitek et al., 2021), these results may be explained from a cognitive load perspective. As the complexity of a task increases, the cognitive resources available for error detection and problem-solving may become overloaded. In the context of code review, higher levels of cyclomatic complexity and nesting depth may therefore impose greater cognitive demands on reviewers, potentially leading them to retract to more shallow processing or unsystematic review (Dehais et al., 2020; Van Der Linden et al., 2003), reducing mental effort but leaving them unable to find nuanced defects hidden in the complex code structures. It may be reflected in the fact that that some reviewers did not spend significantly more time for the review even when complexity increased.

Notably, the uncertainty surrounding the effect of nesting depth on defect detection suggests that the impact of nesting depth may vary depending on additional contextual factors or individual differences among code reviewers. Further research is needed to explore these moderating factors and gain a better understanding of the relationship between nesting depth and defect detection.

5.3 Practical implications

Overall, our findings suggest that code complexity can have a detrimental effect of the quality of code review. Recognising the impact of complexity metrics, such as cyclomatic complexity and nesting depth, on code review effectiveness, organisations can incorporate complexity-aware practices into their code review processes. Taking measures in the form of

checklists and guidelines for maximum acceptable complexity may be a right step but may not be enough. Sometimes it may be unavoidable to have a high degree of complexity in a code change. Some viewpoints argue that certain forms of complexity, such as nesting, can be beneficial in terms of efficiency (Alrasheed & Melton, 2022). These perspectives emphasise that complexity should not solely be seen as an obstacle to overcome but rather as a design choice that balances different factors.

For complex code, companies should therefore consider leveraging multiple reviewers, thereby decreasing the risk of individual reviewers missing critical defects (Thongtanunam et al., 2015; Wang et al., 2021). Furthermore, the variability observed in the effects of complexity on review outcomes highlights the importance of collaboration and knowledge sharing among code reviewers. Creating an environment that promotes collaboration, open discussions, and sharing of expertise can help mitigate the challenges posed by complex code. For example, by using synchronous communication tools for code review, organisations can harness the collective intelligence of many reviewers, where those who seem better equipped to deal with increasing complexity can help those that struggle more. In the long run, this may not only lead to overall improved performance in terms of defect detection ability and time needed for review. It may also reduce the cognitive strain imposed by reviewing complex code (Pascarella et al., 2018).

5.4 Limitations and directions for future research

While this study is the first to show the precise impact of complexity metrics on workload, efficiency, effectiveness, it has several limitations. One significant limitation is the small sample size of only 17 participants. As the study was conducted with professional developers in real-world settings participants had to add the review to their usual workload. Because the general workload was already high and the review of eight code snippets is rather time consuming, voluntary study participation was low. Additionally, as this study was voluntary, it is susceptible to self-selection bias. For example, those that decided to participate may have different characteristics, motivations, or workload compared to those that did not participate. This can introduce bias in the results, particularly if these factors are related to the investigated outcome variables.

These circumstances may not only reduce the statistical power but also generalisability of the findings. Time pressure and knowing that this is only a research project may have additionally led participants to not review the snippets as thoroughly as the would usually do. Such an attempt to save time could have further reduced the generalisability of results and

future research should (1) recruit a larger participant pool and (2) take measures to ensure that participants do not feel pressured by the participation. Yet, results still provide valuable insights as this study's focus on within-subject comparisons mitigates these influences. By comparing the participants' performance and experiences across different levels of complexity, each participant served as their own control, thereby accounting for some of the potential bias.

Another limitation of this study is the fact that it included a limited number of variables. First, it only focused on two complexity metrics, namely cyclomatic complexity and nesting depth. As code can be complex is various ways, other metrics that capture different aspect of code complexity may have confounded the results and should be included in the analysis. While future research should explore additional metrics to provide a more comprehensive picture of their influence on workload and performance, cyclomatic complexity and nesting depth provide a useful starting point as they are widely used in software development.

Second, this study did not include defect type as a variable. Defect types in this study included logic errors, signing errors, out-of-bounds, and data race errors that varied between code snippets. However, different types of defects may have varying degrees of complexity and may impact reviewer performance differently. For example, signing errors may require levels levels of cognitive effort to detect than out-of-bounds errors. In fact, Baum et al. (2019) showed that working memory capacity only affects detection effectiveness of delocalised defects. To gain better insights into their influence on code review performance, future research should consider categorising types of defects and incorporate the type as a factor in the analysis. Alternatively, to gain better insights into the effect of complexity on a certain type of defect, it should be ensured that the type of defect is held constant across snippets. However, this study still provides a good first impression about impact of complexity on the overall ability to detect defects.

Third, this study did not consider code comments. The presence and quality of code comments can greatly influence the perception of code complexity and the reviewer's experience. Well-commented code can provide valuable insights into the logic and structure, potentially reducing the perceived complexity. Conversely, poorly documented code may exacerbate the perceived complexity and add to the cognitive load of the reviewer (Pascarella et al., 2018). The absence of a systematic assessment of code comments in this study may have introduced a confounding factor that could have influenced the results. Future research should include the presence and quality of code comments as an additional variable that may impact the perception of code complexity.

Generally, this study serves as an initial exploration of the relationship between code complexity and review outcomes, providing a foundation for future research to expand the scope and depth of investigation in this area. Subsequent studies should aim to extend the current study by exploring additional factors and variables to enhance our understanding of the complex dynamics underlying code complexity and its impact on review outcomes. This could involve broadening the participant pool to include a wider range of expertise and experience levels, exploring individual factors that may influence the relationship between complexity metrics and code review performance Additionally, investigating different contexts such as programming languages, review tools, software applications, and stages of software development would provide valuable insights into the specific impacts of complexity in these areas and inform tailored complexity management strategies. Investigating the role of team dynamics and collaboration in code review would provide a more comprehensive understanding of how complexity interacts with social factors in the review process. Such modifications to this paradigm would contribute to a more comprehensive understanding of the complex dynamics between code complexity and review outcomes, thereby advancing the field of software engineering and human factors research.

Another limitation of this study is the reliance on subjective measures to assess mental workload. Although subjective measures can be informative, they may be influenced by situational factors other than the variables of concern (Jahedi & Méndez, 2014). Additionally, the NASA-TLX employed in this study uses scales ranging from 0-100 without including anchor points (Hart & Staveland, 1988). This may lead to interpretation and consequently rating differences between participants, making comparisons between individuals more difficult (Cockburn & Gutwin, 2019; Hart, 2006). Future studies should explore other measures of workload to obtain a more comprehensive understanding of workload during code review. Yet, the NASA-TLX is a valuable tool for assessing workload in human factors research, as has been widely used, validated, and applied across various domains (Grier, 2015; Hart, 2006).

Furthermore, due to the remote nature of the study, technical constraints did not allow for a randomisation of snippet order, introducing potential order effects. The fixed order of the code snippets may have influenced participants' workload and performance, consequently confounding the results. As participants progressed through the review tasks, changes in workload and performance may have partially stemmed from the time spent reviewing rather than the investigated changes in code complexity. Future research should therefore randomise the order of snippets to reduce this threat to internal validity.

Future research should further explore the effectiveness of interventions aimed at reducing code complexity or enhancing reviewers' ability to handle complex code. By implementing training programs or tools designed to improve understanding and management of code complexity and comparing them to a control group, researchers can validate their effectiveness, improving the overall code review process.

6 Conclusion

In conclusion, this study examined the impact of code complexity, specifically cyclomatic complexity and nesting depth, on workload and code review performance, including efficiency and effectiveness. Our findings shed light on the relationship between complexity metrics and these measures, providing insights into the challenges and associated implications for code review in software development. By considering the human factors involved in code review, organisations can optimise their development processes and promote more efficient and effective code review practices that put less strain on individual reviewers. It allows for proactive management and mitigation of the challenges code complexity imposes on reviewers, ultimately not only improving reviewer experiences but crucially, also the quality of software systems.

7 References

Alrasheed, H., & Melton, A. (2022). Measuring nesting. *IET Software*, *16*(6), 543–557. https://doi.org/10.1049/sfw2.12069

Antinyan, V., Staron, M., & Sandberg, A. (2017). Evaluating code complexity triggers, use of complexity measures and the influence of code complexity on maintenance time. *Empirical Software Engineering*, *22*(6), 3057–3087. https://doi.org/10.1007/s10664-017-9508-2

Anu, V., Hu, W., Carver, J. C., Walia, G. S., & Bradshaw, G. (2018). Development of a human error taxonomy for software requirements: A systematic literature review. *Information and Software Technology*, *103*, 112–124. https://doi.org/10.1016/j.infsof.2018.06.011

Bacchelli, A., & Bird, C. (2013). Expectations, outcomes, and challenges of modern code review. *2013 35th International Conference on Software Engineering (ICSE)*, 712–721. https://doi.org/10.1109/ICSE.2013.6606617

Barb, A. S., Neill, C. J., Sangwan, R. S., & Piovoso, M. J. (2014). A statistical study of the relevance of lines of code measures in software projects. *Innovations in Systems and Software Engineering*, *10*(4), 243–260. https://doi.org/10.1007/s11334-014-0231-5

Barr, M. (2013). Bookout vs. Toyota. *case No. CJ-2008-7969, District Court of Oklahoma County, http://www. safetyresearch. net/Library/Bookout_v_Toyota_Barr_redacted. pdf, consultado el*, *10*.

Barrouillet, P., & Lecas, J.-F. (1999). Mental Models in Conditional Reasoning and Working Memory. *Thinking & Reasoning*, *5*(4), 289–302. https://doi.org/10.1080/135467899393940

Baum, T., Schneider, K., & Bacchelli, A. (2017). On the Optimal Order of Reading Source Code Changes for Review. *2017 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, 329–340. https://doi.org/10.1109/ICSME.2017.28

Baum, T., Schneider, K., & Bacchelli, A. (2019). Associating working memory capacity and code change ordering with code review performance. *Empirical Software Engineering*, *24*(4), 1762–1798. https://doi.org/10.1007/s10664-018-9676-8

Bavota, G., & Russo, B. (2015). Four eyes are better than two: On the impact of code reviews on software quality. *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, 81–90. https://doi.org/10.1109/ICSM.2015.7332454

Berkowitz, L. (1989). Frustration-aggression hypothesis: Examination and reformulation. *Psychological Bulletin*, *106*(1), 59–73. https://doi.org/10.1037/0033-2909.106.1.59

Bruggen, A. (2015). An empirical investigation of the relationship between workload and performance. *Management Decision*, *53*(10), 2377–2389. https://doi.org/10.1108/MD-02-2015-0063

Bürkner, P.-C. (2017). brms: An R Package for Bayesian Multilevel Models Using Stan. *Journal of Statistical Software*, *80*(1). https://doi.org/10.18637/jss.v080.i01

Calcagno, V., & de Mazancourt, C. (2010). glmulti: An R Package for Easy Automated Model Selection with (Generalized) Linear Models. *Journal of Statistical Software*, *34*(12), 1–29. https://doi.org/10.18637/jss.v034.i12

Chen, C., & Caza, A. (2018, January 22). *Grit, Intrinsic Motivation, and Costly Perseverance: Their Interactive Influence in Problem Solving*.

Cockburn, A., & Gutwin, C. (2019). Anchoring Effects and Troublesome Asymmetric Transfer in Subjective Ratings. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 1–12. https://doi.org/10.1145/3290605.3300592

Curtis, B., Sheppard, S. B., Milliman, P., Borst, M. A., & Love, T. (1979). Measuring the Psychological Complexity of Software Maintenance Tasks with the Halstead and McCabe Metrics. *IEEE Transactions on Software Engineering*, *SE-5*(2), 96–104. https://doi.org/10.1109/TSE.1979.234165

de Winter, J. C. F. (2014). Controversy in human factors constructs and the explosive use of the NASA-TLX: A measurement perspective. *Cognition, Technology & Work*, *16*(3), 289–297. https://doi.org/10.1007/s10111-014-0275-1

Dehais, F., Lafont, A., Roy, R., & Fairclough, S. (2020). A Neuroergonomics Approach to Mental Workload, Engagement and Human Performance. *Frontiers in Neuroscience*, *14*, 268. https://doi.org/10.3389/fnins.2020.00268

Ebert, F., Castor, F., Novielli, N., & Serebrenik, A. (2021). An exploratory study on confusion in code reviews. *Empirical Software Engineering*, *26*(1), 12. https://doi.org/10.1007/s10664-020-09909-5

Fagan, M. E. (1976). Design and code inspections to reduce errors in program development. *IBM Systems Journal*, *15*(3), 182–211. https://doi.org/10.1147/sj.153.0182

Fan, J., & Smith, A. P. (2017). The Impact of Workload and Fatigue on Performance. In L. Longo & M. C. Leva (Eds.), *Human Mental Workload: Models and Applications* (Vol. 726, pp. 90–105). Springer International Publishing. https://doi.org/10.1007/978-3-319-61061-0_6

Grier, R. A. (2015). How High is High? A Meta-Analysis of NASA-TLX Global Workload Scores. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *59*(1), 1727–1731. https://doi.org/10.1177/1541931215591373

Harrison, W. A., & Magel, K. I. (1981). A complexity measure based on nesting level. *ACM SIGPLAN Notices*, *16*(3), 63–74. https://doi.org/10.1145/947825.947829

Hart, S. G. (2006). Nasa-Task Load Index (NASA-TLX); 20 Years Later. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, *50*(9), 904–908. https://doi.org/10.1177/154193120605000909

Hart, S. G., & Staveland, L. E. (1988). Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In *Advances in Psychology* (Vol. 52, pp. 139–183). Elsevier. https://doi.org/10.1016/S0166-4115(08)62386-9

Hijazi, H., Duraes, J., Couceiro, R., Castelhano, J., Barbosa, R., Medeiros, J., Castelo-Branco, M., De Carvalho, P., & Madeira, H. (2023). Quality Evaluation of Modern Code Reviews Through Intelligent Biometric Program Comprehension. *IEEE Transactions on Software Engineering*, *49*(2), 626–645. https://doi.org/10.1109/TSE.2022.3158543

Huang, F., Liu, B., & Huang, B. (2012). *A Taxonomy System to Identify Human Error Causes for Software Defects*. https://doi.org/10.13140/2.1.4528.5445

Huang, F., Liu, B., Wang, S., & Li, Q. (2015). The impact of software process consistency on residual defects: Impact of Software Process Consistency on Residual Defects. *Journal of Software: Evolution and Process*, *27*(9), 625–646. https://doi.org/10.1002/smr.1717

Jahedi, S., & Méndez, F. (2014). On the advantages and disadvantages of subjective measures. *Journal of Economic Behavior & Organization*, *98*, 97–114. https://doi.org/10.1016/j.jebo.2013.12.016

Khoshnoud, F., Nasab, A. R., Toudeji, Z., & Sami, A. (2022). Which bugs are missed in code reviews: An empirical study on SmartSHARK dataset. *Proceedings of the 19th International Conference on Mining Software Repositories*, 137–141. https://doi.org/10.1145/3524842.3527997

Kononenko, O., Baysal, O., & Godfrey, M. W. (2016). Code review quality: How developers see it. *Proceedings of the 38th International Conference on Software Engineering*, 1028–1038. https://doi.org/10.1145/2884781.2884840

Leveson, N. G. (2011). Applying systems thinking to analyze and learn from events. *Safety Science*, *49*(1), 55–64. https://doi.org/10.1016/j.ssci.2009.12.021

McCabe, T. J. (1976). A Complexity Measure. *IEEE Transactions on Software Engineering*, *SE-2*(4), 308–320. https://doi.org/10.1109/TSE.1976.233837

McIntosh, S., Kamei, Y., Adams, B., & Hassan, A. E. (2014). The impact of code review coverage and code review participation on software quality: A case study of the qt, VTK, and ITK projects. *Proceedings of the 11th Working Conference on Mining Software Repositories*, 192–201. https://doi.org/10.1145/2597073.2597076

Nuñez-Varela, A. S., Pérez-Gonzalez, H. G., Martínez-Perez, F. E., & Soubervielle-Montalvo, C. (2017). Source code metrics: A systematic mapping study. *Journal of Systems and Software*, *128*, 164–197. https://doi.org/10.1016/j.jss.2017.03.044

Pascarella, L., Spadini, D., Palomba, F., Bruntink, M., & Bacchelli, A. (2018). Information Needs in Contemporary Code Review. *Proceedings of the ACM on Human-Computer Interaction*, *2*(CSCW), 1–27. https://doi.org/10.1145/3274404

Peitek, N., Apel, S., Parnin, C., Brechmann, A., & Siegmund, J. (2021). Program Comprehension and Code Complexity Metrics: An fMRI Study. *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, 524–536. https://doi.org/10.1109/ICSE43902.2021.00056

Reason, J. (1990). *Human Error* (1st ed.). Cambridge University Press. https://doi.org/10.1017/CBO9781139062367

Rigby, P. C., & Bird, C. (2013). Convergent contemporary software peer review practices. *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*, 202–212. https://doi.org/10.1145/2491411.2491444

Sadowski, C., Söderberg, E., Church, L., Sipko, M., & Bacchelli, A. (2018). Modern code review: A case study at google. *Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice*, 181–190. https://doi.org/10.1145/3183519.3183525

Sharif, B., Falcone, M., & Maletic, J. I. (2012). An eye-tracking study on the role of scan time in finding source code defects. *Proceedings of the Symposium on Eye Tracking Research and Applications*, 381–384. https://doi.org/10.1145/2168556.2168642

Shepperd, M. (1988). A critique of cyclomatic complexity as a software metric. *Software Engineering Journal*, *3*(2), 30. https://doi.org/10.1049/sej.1988.0003

Shin, Y., & Williams, L. (2008). An empirical model to predict security vulnerabilities using code complexity metrics. *Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, 315–317. https://doi.org/10.1145/1414004.1414065

Stein, M., Riedl, J., Harner, S. J., & Mashayekhi, V. (1997). A case study of distributed, asynchronous software inspection. *Proceedings of the 19th International Conference on Software Engineering - ICSE '97*, 107–117. https://doi.org/10.1145/253228.253250

Suleman Sarwar, M. M., Shahzad, S., & Ahmad, I. (2013). Cyclomatic complexity: The nesting problem. *Eighth International Conference on Digital Information Management (ICDIM 2013)*, 274–279. https://doi.org/10.1109/ICDIM.2013.6693981

Sweller, J. (1988). Cognitive Load During Problem Solving: Effects on Learning. *Cognitive Science*, *12*(2), 257–285. https://doi.org/10.1207/s15516709cog1202_4

Sweller, J., Ayres, P., & Kalyuga, S. (2011a). *Cognitive Load Theory*. Springer New York. https://doi.org/10.1007/978-1-4419-8126-4

Sweller, J., Ayres, P., & Kalyuga, S. (2011b). Measuring Cognitive Load. In J. Sweller, P. Ayres, & S. Kalyuga, *Cognitive Load Theory* (pp. 71–85). Springer New York. https://doi.org/10.1007/978-1-4419-8126-4_6

Thongtanunam, P., McIntosh, S., Hassan, A. E., & Iida, H. (2015). Investigating Code Review Practices in Defective Files: An Empirical Study of the Qt System. *2015 IEEE/ACM 12th Working Conference on Mining Software Repositories*, 168–179. https://doi.org/10.1109/MSR.2015.23

Van Der Linden, D., Frese, M., & Sonnentag, S. (2003). The Impact of Mental Fatigue on Exploration in a Complex Computer Task: Rigidity and Loss of Systematic Strategies. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *45*(3), 483–494. https://doi.org/10.1518/hfes.45.3.483.27256

Wang, D., Wang, Q., Wang, J., & Shi, L. (2021). Accept or Not? An Empirical Study on Analyzing the Factors that Affect the Outcomes of Modern Code Review? *2021 IEEE 21st International Conference on Software Quality, Reliability and Security (QRS)*, 946–955. https://doi.org/10.1109/QRS54544.2021.00104

Wang, Y., Yijun, L., Collins, M., & Liu, P. (2008). Process improvement of peer code review and behavior analysis of its participants. *ACM SIGCSE Bulletin*, *40*(1), 107–111. https://doi.org/10.1145/1352322.1352171

Wickens, C. D. (Ed.). (2014). *An introduction to human factors engineering* (2. ed., Pearson new internat. ed). Pearson Education.

Wickens, C. D., Helton, W. S., Hollands, J. G., & Banbury, S. (2021). *Engineering Psychology and Human Performance* (5th ed.). Routledge. https://doi.org/10.4324/9781003177616

Zuse, H. (1991). *Software Complexity: Measures and Methods*. De Gruyter. https://doi.org/10.1515/9783110866087

8 Appendices

8.1 Appendix A: Questionnaire

Microsoft forms

Human Factors in Software Engineering

Dear Participant,

You are being invited to participate in a research study titled “The Influence of Code Complexity on Review Efficiency, Effectiveness and Workload in Embedded Software Development”. The study is being conducted by Gina Jahncke from the Faculty of Behavioural, Management and Social Sciences (BMS) at the University of Twente and is part of a master thesis in the field of human factors engineering. It has been reviewed and approved by the BMS Ethics Committee/domain Humanities & Social Sciences.

As a participant, you will be asked to complete an online survey that includes questions about your background and experience in code review. You will then be asked to review eight code snippets and provide feedback on your experience. Your participation in this study will contribute to the advancement of knowledge in human factors engineering and embedded software development. Before deciding to participate, please take the time to read the following information.

Voluntary participation:
Participation in this study is entirely voluntary, and you may withdraw at any time. You may decline to answer any questions or discontinue participation at any time.

Confidentiality:
We believe there are no known risks associated with this research study; however, as with any online related activity the risk of a breach is always possible. To protect your data, your participation and responses will be kept strictly confidential. We will not ask for any personally identifiable data, instead, a unique identifier will be assigned to each participant to ensure anonymity. All data will be used for academic purposes only and will be stored on secure servers. Information that would make it possible to identify you or any other participant will never be included in any report or publication.

Contact information:
If you have any questions about this study, please contact the investigator Gina Jahncke at gina.jahncke@vector.com.

Consent:
By proceeding with this survey, you acknowledge that you have read and understood the

information provided in this informed consent form. Your consent to participate in this study is implied by completing the survey.

Background Information

1. Years of experience with professional software development

   Der Wert muss eine Zahl sein.

2. In your life, how many **months** have you predominantly programmed using C?

   Der Wert muss eine Zahl sein.

3. Approximately how many lines of code have you written using C each month?

   Der Wert muss eine Zahl sein.

4. How many **months** have you already been doing code review at Vector?

   Der Wert muss eine Zahl sein.

5. Approximately how many lines of code do you review each month?

Der Wert muss eine Zahl sein.

6. How many hours have you already worked today?

Der Wert muss eine Zahl sein.

7. How confident are you in your ability to detect mistakes?

- ◯ Extremely confident
- ◯ Somewhat confident
- ◯ Neutral
- ◯ Somewhat not confident
- ◯ Extremely not confident

8. How do you perceive your current level of fitness?

- ◯ Not fatigued at all
- ◯ A little fatigued
- ◯ Moderately fatigued
- ◯ Very fatigued
- ◯ Total fatigue / complete exhaustion

File Download

Before downloading the file, **please read this information carefully**:

The file contains eight folders. Each folder holds…

1. one code snippet to review (they may or may not include artificially produced faults),
2. the report sheet, and
3. a workload questionnaire.

Please **go through the folders from top to bottom** by completing the following steps:

1. Review the snippet in the first folder.
2. Fill in the report excel.
   Note: You don't have to fill in the checklist! Only fill in the findings and the effort! ***Please measure your review time with a stopwatch and report it in minutes, not hours***.
3. Fill in the workload questionnaire.

Once you have completed these three steps for the first folder, you will move on to the next folder, redo the same three steps and so on.

Once you are done with reviewing the code snippets, please upload the file back to the MS forms! The study is completed once you have **uploaded the zip file** and completed the questionnaire.

Please click on this link to download the file and complete the task as explained above: https://vgroup-my.sharepoint.com/:u:/g/personal/gina_jahncke_vector_com/EUgi-bvyVZVKIbQ4RVLO_tkBdjpgwsMmvt3p8Sz8g22McQ?e=4OlNQM

File Upload

9. Please upload the completed review zip file here (please **rename the file to have a ".pdf" ending**, as MS does not allow the upload of zip files). *

   Important note: we have taken external measures to ensure that the file upload remains completely anonymous. Your file will never be associated with your name.

   Datei hochladen

   Limit für Dateianzahl: 1 Größenlimit für eine einzelne Datei: 1GB Zulässige Dateitypen: Word, Excel, PPT, PDF, Bild, Video, Audio

8.2 Appendix B: R syntax

1 Install & Load Packages Needed for Analysis

```
install.packages("knitr")
install.packages("stringr")
install.packages("devtools")
install.packages("haven")
install.packages("broom.mixed")
install.packages("rstanarm")
install.packages("brms")
install.packages("ggimg")
install.packages("ggplot2")
install.packages("bayr")
install.packages("dplyr")
install.packages("GGally")
install.packages("patchwork")

library(tidyverse)

## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──
## ✓ ggplot2 3.4.2     ✓ purrr   0.3.5
## ✓ tibble  3.2.1     ✓ dplyr   1.1.2
## ✓ tidyr   1.2.1     ✓ stringr 1.5.0
## ✓ readr   2.1.3     ✓ forcats 0.5.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()

library(rstanarm)

## Loading required package: Rcpp
## This is rstanarm version 2.21.4
## - See https://mc-stan.org/rstanarm/articles/priors for changes to default priors!
## - Default priors may change, so it's safest to specify priors, even if equivalent to the
defaults.
## - For execution on a local, multicore CPU with excess RAM we recommend calling
##   options(mc.cores = parallel::detectCores())

library(brms)

## Loading 'brms' package (version 2.19.0). Useful instructions
## can be found by typing help('brms'). A more detailed introduction
## to the package is available through vignette('brms_overview').
##
## Attaching package: 'brms'
##
## The following objects are masked from 'package:rstanarm':
##
##     dirichlet, exponential, get_y, lasso, ngrps
##
## The following object is masked from 'package:stats':
##
##     ar

options(mc.cores = 4)
library(bayr)

## Registered S3 methods overwritten by 'bayr':
##   method          from
##   coef.brmsfit    brms
```

```
##   coef.stanreg    rstanarm
##   predict.brmsfit brms
##   predict.stanreg rstanarm
##
## Attaching package: 'bayr'
##
## The following objects are masked from 'package:brms':
##
##     fixef, ranef
##
## The following objects are masked from 'package:rstanarm':
##
##     fixef, ranef
##
## The following object is masked from 'package:tidyr':
##
##     expand_grid

library(GGally)

## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2

library(patchwork)
```

2 Reading & Cleaning Data

2.1 Import Files

```
D_Part <-  ## participant-level variables
  read.csv("MS_results.csv", sep = ';')%>%
  rename("Part" = "ID",
         "sd_experience_yrs" = "Years.of.experience.with.professional.software.development"
,
         "c_experience_months" = "In.your.life..how.many.months.have.you.predominantly.prog
rammed.using.C.",
         "loc_written_pm" = "Approximately.how.many.lines.of.code.have.you.written.using.C.
each.month.",
         "review_months" = "How.many.months.have.you.already.been.doing.code.review.at.Vect
or.",
         "loc_reviewed_pm" = "Approximately.how.many.lines.of.code.do.you.review.each.month
.",
         "working_hrs" = "How.many.hours.have.you.already.worked.today.",
         "Confidence" = "How.confident.are.you.in.your.ability.to.detect.mistakes.",
         "Fitness"= "How.do.you.perceive.your.current.level.of.fitness.")%>%
  select(-Startzeit, -Fertigstellungszeit, -E.Mail, -Name, -Please.upload.the.completed.rev
iew.zip.file.here..please.rename.the.file.to.have.a...pdf..ending..as.MS.does.not.allow.the
.upload.of.zip.files..)
  print(D_Part)
```

```
##    Part sd_experience_yrs c_experience_months loc_written_pm review_months
## 1     4                20                 228            200            10
## 2     5                 6                  24             20            72
## 3     6                21                 252            100           240
## 4     7                15                  60            500            60
## 5     8                 5                  36            200            48
## 6     9                27                 270            200           180
## 7    10                10                  98            200           108
## 8    11                 5                 100            600            25
## 9    12                10                 120            100            30
## 10   13                20                 180            500             5
## 11   14                 3                  32           1000            34
## 12   15                 7                 120            500            72
## 13   16                 4                  84           3000            42
```

```
## 14   17                4                  96            100            46
## 15   18               20                 245             NA           244
## 16   19                6                  60            300            60
## 17   20               12                 100            100            60
##    loc_reviewed_pm working_hrs             Confidence             Fitness
## 1              100           1 Somewhat not confident Not fatigued at all
## 2              500           1     Somewhat confident   A little fatigued
## 3              100           2                Neutral   A little fatigued
## 4             5000           2     Somewhat confident Moderately fatigued
## 5               NA           0     Somewhat confident Moderately fatigued
## 6              500           6                Neutral Not fatigued at all
## 7             1000           5     Somewhat confident Moderately fatigued
## 8               80           2     Somewhat confident Not fatigued at all
## 9              300           4    Extremely confident   A little fatigued
## 10             300           0     Somewhat confident Not fatigued at all
## 11            1000           4     Somewhat confident   A little fatigued
## 12             400           6     Somewhat confident Moderately fatigued
## 13            5000           1     Somewhat confident   A little fatigued
## 14              50           0     Somewhat confident Not fatigued at all
## 15              NA           0     Somewhat confident Not fatigued at all
## 16              50           5     Somewhat confident   A little fatigued
## 17             100           2     Somewhat confident Moderately fatigued

D_Exp <- # experimental results
  read.csv("Snippet_results.csv", sep = ';')%>%
  rename("Part" = "ID", "cyclomatic_complexity" = "CC", "nesting_depth" = "ND", "lines_of_c
ode" = "LOC", "Fault_detection" = "Fault.Detection", "mental_demand" = "Mental.Demand", "te
mporal_demand" = "Temporal.Demand", "effort" = "Effort", "frustration" = "Frustration")%>%
  select(-Comments)
  print(D_Exp)

##     Part Snippet cyclomatic_complexity nesting_depth lines_of_code ToT
## 1      4       1                     3             1            49   5
## 2      4       2                     6             4            96  10
## 3      4       3                     9             7           125   5
## 4      4       4                    12             2           117  10
## 5      4       5                    15             5           222  15
## 6      4       6                    18             8           273  NA
## 7      4       7                    21             3           183  15
## 8      4       8                    27             6           370  NA
## 9      5       1                     3             1            49   5
## 10     5       2                     6             4            96  10
## 11     5       3                     9             7           125  10
## 12     5       4                    12             2           117   5
## 13     5       5                    15             5           222  21
## 14     5       6                    18             8           273  13
## 15     5       7                    21             3           183   5
## 16     5       8                    27             6           370   6
## 17     6       1                     3             1            49   5
## 18     6       2                     6             4            96   4
## 19     6       3                     9             7           125   4
## 20     6       4                    12             2           117   4
## 21     6       5                    15             5           222   6
## 22     6       6                    18             8           273   6
## 23     6       7                    21             3           183   7
## 24     6       8                    27             6           370   6
## 25     7       1                     3             1            49   7
## 26     7       2                     6             4            96  12
## 27     7       3                     9             7           125  22
## 28     7       4                    12             2           117  11
## 29     7       5                    15             5           222   9
## 30     7       6                    18             8           273  37
## 31     7       7                    21             3           183  15
## 32     7       8                    27             6           370  NA
## 33     8       1                     3             1            49   5
## 34     8       2                     6             4            96  10
```

```
## 35     8       3                  9              7          125  12
## 36     8       4                 12              2          117   5
## 37     8       5                 15              5          222  20
## 38     8       6                 18              8          273  15
## 39     8       7                 21              3          183   5
## 40     8       8                 27              6          370  20
## 41     9       1                  3              1           49   3
## 42     9       2                  6              4           96   3
## 43     9       3                  9              7          125   4
## 44     9       4                 12              2          117   8
## 45     9       5                 15              5          222   8
## 46     9       6                 18              8          273  NA
## 47     9       7                 21              3          183  NA
## 48     9       8                 27              6          370  NA
## 49    10       1                  3              1           49  14
## 50    10       2                  6              4           96  11
## 51    10       3                  9              7          125  15
## 52    10       4                 12              2          117  12
## 53    10       5                 15              5          222  24
## 54    10       6                 18              8          273  40
## 55    10       7                 21              3          183  17
## 56    10       8                 27              6          370  42
## 57    11       1                  3              1           49  10
## 58    11       2                  6              4           96  20
## 59    11       3                  9              7          125  35
## 60    11       4                 12              2          117  15
## 61    11       5                 15              5          222   5
## 62    11       6                 18              8          273   5
## 63    11       7                 21              3          183  15
## 64    11       8                 27              6          370   5
## 65    12       1                  3              1           49  10
## 66    12       2                  6              4           96  20
## 67    12       3                  9              7          125  20
## 68    12       4                 12              2          117  15
## 69    12       5                 15              5          222  30
## 70    12       6                 18              8          273  10
## 71    12       7                 21              3          183  10
## 72    12       8                 27              6          370  15
## 73    13       1                  3              1           49   8
## 74    13       2                  6              4           96   7
## 75    13       3                  9              7          125  17
## 76    13       4                 12              2          117   8
## 77    13       5                 15              5          222  20
## 78    13       6                 18              8          273  22
## 79    13       7                 21              3          183  16
## 80    13       8                 27              6          370  16
## 81    14       1                  3              1           49   7
## 82    14       2                  6              4           96   8
## 83    14       3                  9              7          125  12
## 84    14       4                 12              2          117  13
## 85    14       5                 15              5          222  12
## 86    14       6                 18              8          273  25
## 87    14       7                 21              3          183   4
## 88    14       8                 27              6          370  20
## 89    15       1                  3              1           49  13
## 90    15       2                  6              4           96  28
## 91    15       3                  9              7          125  28
## 92    15       4                 12              2          117  13
## 93    15       5                 15              5          222  30
## 94    15       6                 18              8          273  24
## 95    15       7                 21              3          183  14
## 96    15       8                 27              6          370  25
## 97    16       1                  3              1           49   9
## 98    16       2                  6              4           96  12
## 99    16       3                  9              7          125  13
## 100   16       4                 12              2          117  12
```

```
## 101   16       5                   15             5           222  30
## 102   16       6                   18             8           273  15
## 103   16       7                   21             3           183  10
## 104   16       8                   27             6           370  NA
## 105   17       1                    3             1            49   8
## 106   17       2                    6             4            96  15
## 107   17       3                    9             7           125  16
## 108   17       4                   12             2           117   8
## 109   17       5                   15             5           222  10
## 110   17       6                   18             8           273  11
## 111   17       7                   21             3           183  10
## 112   17       8                   27             6           370  11
## 113   18       1                    3             1            49   7
## 114   18       2                    6             4            96   9
## 115   18       3                    9             7           125  28
## 116   18       4                   12             2           117  14
## 117   18       5                   15             5           222  30
## 118   18       6                   18             8           273  25
## 119   18       7                   21             3           183  20
## 120   18       8                   27             6           370  13
## 121   19       1                    3             1            49   3
## 122   19       2                    6             4            96   7
## 123   19       3                    9             7           125  13
## 124   19       4                   12             2           117   7
## 125   19       5                   15             5           222   9
## 126   19       6                   18             8           273  18
## 127   19       7                   21             3           183   6
## 128   19       8                   27             6           370  12
## 129   20       1                    3             1            49   5
## 130   20       2                    6             4            96   5
## 131   20       3                    9             7           125   8
## 132   20       4                   12             2           117   9
## 133   20       5                   15             5           222  14
## 134   20       6                   18             8           273  15
## 135   20       7                   21             3           183  15
## 136   20       8                   27             6           370  40
##     Fault_detection mental_demand temporal_demand effort frustration
## 1                 0             NA              NA     NA          NA
## 2                 1             NA              NA     NA          NA
## 3                 1             NA              NA     NA          NA
## 4                 0             NA              NA     NA          NA
## 5                 1             NA              NA     NA          NA
## 6                 0             NA              NA     NA          NA
## 7                 0             NA              NA     NA          NA
## 8                 0             NA              NA     NA          NA
## 9                 0             30              60     35          45
## 10                1             60              45     65          45
## 11                1             75              30     75          40
## 12                1             25              20     25          20
## 13                0             90              30     80          80
## 14                0             65              30     60          35
## 15                1             35              25     40          25
## 16                0             45              30     30          85
## 17                0             25              15     20          45
## 18                1             15              25     15          45
## 19                0             30              35     35          55
## 20                0             30              20     20          60
## 21                0             55              20     45          90
## 22                0             55              45     60          90
## 23                0             60              60     45          95
## 24                0             65              20     20         100
## 25                0             35              15     60          40
## 26                1             70              25     70          35
## 27                0             85              25     70          75
## 28                1             30              20     30          35
## 29                0             95              20    100         100
```

```
## 30                0             90              20     85          75
## 31                1             NA              NA     NA          NA
## 32                0             90              20     95          60
## 33                0             50              25     50          25
## 34                1             50              20     50          20
## 35                1             50              55     55          50
## 36                1             50              20     50          20
## 37                0             75              75     75          75
## 38                0             65              60     60          60
## 39                1             50              50     30          30
## 40                0             70              60     60          60
## 41                0             25              50     10          10
## 42                1             20              50     15          15
## 43                1             15              50     20          10
## 44                1             30              75     30          15
## 45                0             65              60     45          50
## 46               NA             NA              NA     NA          NA
## 47               NA             NA              NA     NA          NA
## 48               NA             NA              NA     NA          NA
## 49                1             10              55     15          65
## 50                1             15              25     15          15
## 51                1             15              15     10          10
## 52                1             20              25     20          25
## 53                1             85              65     75          80
## 54                0             80              60     85          85
## 55                0             80              40     70          55
## 56                1             90              65     85          85
## 57                1             25              45     55           5
## 58                1             60              55     40          10
## 59                1             70              35     70          10
## 60                0             65              35     35          35
## 61                0             15               5      5          80
## 62                0             25              15     10          20
## 63                0             60              60     60          80
## 64                0             10              10     10          45
## 65                1             65              60     55          65
## 66                1             65              30     60          45
## 67                0             70              55     70          75
## 68                0             25              15     35          10
## 69                1             90              50     90          95
## 70                0             35              40     55          65
## 71                0             65              40     55          65
## 72                0             55              40     60          70
## 73                0             15               5     15           5
## 74                1             15               5     15           5
## 75                1             30              15     30          10
## 76                1             15              15     15           5
## 77                0             70              35     70          70
## 78                0             70              35     70          55
## 79                0             55              25     60          30
## 80                0             75              60     75          80
## 81                0             50              50     50          60
## 82                0             60              55     60          55
## 83                1             65              65     65          65
## 84                1             70              70     75          90
## 85                0             85              90     95         100
## 86                0             85              50     90          95
## 87                1             80              80     85          85
## 88                1             45              50     55          50
## 89                0             50              40     30          50
## 90                1             70              50     70          40
## 91                1             60              35     40          30
## 92                1             25              40     20          15
## 93                1             85              75     80          90
## 94                0             65              50     60          45
## 95                1             65              45     65          35
```

```
## 96                0            70              45     60          35
## 97                1            45              35     40          40
## 98                1            45              45     40          35
## 99                1            55              50     45          50
## 100               1            40              40     45          45
## 101               0            80              90     50          95
## 102               0            95              90     65          95
## 103               0            75              80     20          45
## 104              NA            NA              NA     NA          NA
## 105               0            30              70     15          10
## 106               0            60              25     45          15
## 107               1            35              20     20          10
## 108               1            60              70     40          60
## 109               0            70              85     90          90
## 110               0            65              55     45          30
## 111               1            35              20     30          10
## 112               0            75              60     80          70
## 113               1            NA              NA     NA          NA
## 114               1            NA              NA     NA          NA
## 115               0            NA              NA     NA          NA
## 116               1            NA              NA     NA          NA
## 117               0            NA              NA     NA          NA
## 118               0            NA              NA     NA          NA
## 119               1            NA              NA     NA          NA
## 120               0            NA              NA     NA          NA
## 121               0            15              75     20          70
## 122               0            20              20     25          15
## 123               1            40              15     35          20
## 124               1            20              15     25          10
## 125               0            85              15     95         100
## 126               0            70              45     70          80
## 127               1            20              15     25          15
## 128               0            30              35     55          45
## 129               0            30              10     15          15
## 130               1            35              10     10          15
## 131               1            40              15     20          15
## 132               1            40              20     25          20
## 133               0            80              65     80          85
## 134               0            70              60     70          70
## 135               1            65              60     65          60
## 136               1            75              65     75          75
```

```
D_Snips <- D_Exp %>% ## extracting a Snippet-level table
  distinct(Snippet, cyclomatic_complexity, nesting_depth, lines_of_code)
  print(D_Snips)
```

```
##   Snippet cyclomatic_complexity nesting_depth lines_of_code
## 1       1                     3             1            49
## 2       2                     6             4            96
## 3       3                     9             7           125
## 4       4                    12             2           117
## 5       5                    15             5           222
## 6       6                    18             8           273
## 7       7                    21             3           183
## 8       8                    27             6           370
```

3 Join Data

```
D_1 <-
  D_Exp %>%
  left_join(D_Part, by = "Part") %>%
  mutate(Part = as.character(Part),
         Snippet = as.character(Snippet))

D_1
```

```
##     Part Snippet cyclomatic_complexity nesting_depth lines_of_code ToT
## 1      4       1                     3             1            49   5
## 2      4       2                     6             4            96  10
## 3      4       3                     9             7           125   5
## 4      4       4                    12             2           117  10
## 5      4       5                    15             5           222  15
## 6      4       6                    18             8           273  NA
## 7      4       7                    21             3           183  15
## 8      4       8                    27             6           370  NA
## 9      5       1                     3             1            49   5
## 10     5       2                     6             4            96  10
## 11     5       3                     9             7           125  10
## 12     5       4                    12             2           117   5
## 13     5       5                    15             5           222  21
## 14     5       6                    18             8           273  13
## 15     5       7                    21             3           183   5
## 16     5       8                    27             6           370   6
## 17     6       1                     3             1            49   5
## 18     6       2                     6             4            96   4
## 19     6       3                     9             7           125   4
## 20     6       4                    12             2           117   4
## 21     6       5                    15             5           222   6
## 22     6       6                    18             8           273   6
## 23     6       7                    21             3           183   7
## 24     6       8                    27             6           370   6
## 25     7       1                     3             1            49   7
## 26     7       2                     6             4            96  12
## 27     7       3                     9             7           125  22
## 28     7       4                    12             2           117  11
## 29     7       5                    15             5           222   9
## 30     7       6                    18             8           273  37
## 31     7       7                    21             3           183  15
## 32     7       8                    27             6           370  NA
## 33     8       1                     3             1            49   5
## 34     8       2                     6             4            96  10
## 35     8       3                     9             7           125  12
## 36     8       4                    12             2           117   5
## 37     8       5                    15             5           222  20
## 38     8       6                    18             8           273  15
## 39     8       7                    21             3           183   5
## 40     8       8                    27             6           370  20
## 41     9       1                     3             1            49   3
## 42     9       2                     6             4            96   3
## 43     9       3                     9             7           125   4
## 44     9       4                    12             2           117   8
## 45     9       5                    15             5           222   8
## 46     9       6                    18             8           273  NA
## 47     9       7                    21             3           183  NA
## 48     9       8                    27             6           370  NA
## 49    10       1                     3             1            49  14
## 50    10       2                     6             4            96  11
## 51    10       3                     9             7           125  15
## 52    10       4                    12             2           117  12
## 53    10       5                    15             5           222  24
## 54    10       6                    18             8           273  40
## 55    10       7                    21             3           183  17
## 56    10       8                    27             6           370  42
## 57    11       1                     3             1            49  10
## 58    11       2                     6             4            96  20
## 59    11       3                     9             7           125  35
## 60    11       4                    12             2           117  15
## 61    11       5                    15             5           222   5
## 62    11       6                    18             8           273   5
## 63    11       7                    21             3           183  15
## 64    11       8                    27             6           370   5
## 65    12       1                     3             1            49  10
```

```
## 66    12       2                  6              4           96  20
## 67    12       3                  9              7          125  20
## 68    12       4                 12              2          117  15
## 69    12       5                 15              5          222  30
## 70    12       6                 18              8          273  10
## 71    12       7                 21              3          183  10
## 72    12       8                 27              6          370  15
## 73    13       1                  3              1           49   8
## 74    13       2                  6              4           96   7
## 75    13       3                  9              7          125  17
## 76    13       4                 12              2          117   8
## 77    13       5                 15              5          222  20
## 78    13       6                 18              8          273  22
## 79    13       7                 21              3          183  16
## 80    13       8                 27              6          370  16
## 81    14       1                  3              1           49   7
## 82    14       2                  6              4           96   8
## 83    14       3                  9              7          125  12
## 84    14       4                 12              2          117  13
## 85    14       5                 15              5          222  12
## 86    14       6                 18              8          273  25
## 87    14       7                 21              3          183   4
## 88    14       8                 27              6          370  20
## 89    15       1                  3              1           49  13
## 90    15       2                  6              4           96  28
## 91    15       3                  9              7          125  28
## 92    15       4                 12              2          117  13
## 93    15       5                 15              5          222  30
## 94    15       6                 18              8          273  24
## 95    15       7                 21              3          183  14
## 96    15       8                 27              6          370  25
## 97    16       1                  3              1           49   9
## 98    16       2                  6              4           96  12
## 99    16       3                  9              7          125  13
## 100   16       4                 12              2          117  12
## 101   16       5                 15              5          222  30
## 102   16       6                 18              8          273  15
## 103   16       7                 21              3          183  10
## 104   16       8                 27              6          370  NA
## 105   17       1                  3              1           49   8
## 106   17       2                  6              4           96  15
## 107   17       3                  9              7          125  16
## 108   17       4                 12              2          117   8
## 109   17       5                 15              5          222  10
## 110   17       6                 18              8          273  11
## 111   17       7                 21              3          183  10
## 112   17       8                 27              6          370  11
## 113   18       1                  3              1           49   7
## 114   18       2                  6              4           96   9
## 115   18       3                  9              7          125  28
## 116   18       4                 12              2          117  14
## 117   18       5                 15              5          222  30
## 118   18       6                 18              8          273  25
## 119   18       7                 21              3          183  20
## 120   18       8                 27              6          370  13
## 121   19       1                  3              1           49   3
## 122   19       2                  6              4           96   7
## 123   19       3                  9              7          125  13
## 124   19       4                 12              2          117   7
## 125   19       5                 15              5          222   9
## 126   19       6                 18              8          273  18
## 127   19       7                 21              3          183   6
## 128   19       8                 27              6          370  12
## 129   20       1                  3              1           49   5
## 130   20       2                  6              4           96   5
## 131   20       3                  9              7          125   8
```

```
## 132   20       4                   12             2           117   9
## 133   20       5                   15             5           222  14
## 134   20       6                   18             8           273  15
## 135   20       7                   21             3           183  15
## 136   20       8                   27             6           370  40
##     Fault_detection mental_demand temporal_demand effort frustration
## 1                 0             NA              NA     NA          NA
## 2                 1             NA              NA     NA          NA
## 3                 1             NA              NA     NA          NA
## 4                 0             NA              NA     NA          NA
## 5                 1             NA              NA     NA          NA
## 6                 0             NA              NA     NA          NA
## 7                 0             NA              NA     NA          NA
## 8                 0             NA              NA     NA          NA
## 9                 0             30              60     35          45
## 10                1             60              45     65          45
## 11                1             75              30     75          40
## 12                1             25              20     25          20
## 13                0             90              30     80          80
## 14                0             65              30     60          35
## 15                1             35              25     40          25
## 16                0             45              30     30          85
## 17                0             25              15     20          45
## 18                1             15              25     15          45
## 19                0             30              35     35          55
## 20                0             30              20     20          60
## 21                0             55              20     45          90
## 22                0             55              45     60          90
## 23                0             60              60     45          95
## 24                0             65              20     20         100
## 25                0             35              15     60          40
## 26                1             70              25     70          35
## 27                0             85              25     70          75
## 28                1             30              20     30          35
## 29                0             95              20    100         100
## 30                0             90              20     85          75
## 31                1             NA              NA     NA          NA
## 32                0             90              20     95          60
## 33                0             50              25     50          25
## 34                1             50              20     50          20
## 35                1             50              55     55          50
## 36                1             50              20     50          20
## 37                0             75              75     75          75
## 38                0             65              60     60          60
## 39                1             50              50     30          30
## 40                0             70              60     60          60
## 41                0             25              50     10          10
## 42                1             20              50     15          15
## 43                1             15              50     20          10
## 44                1             30              75     30          15
## 45                0             65              60     45          50
## 46               NA             NA              NA     NA          NA
## 47               NA             NA              NA     NA          NA
## 48               NA             NA              NA     NA          NA
## 49                1             10              55     15          65
## 50                1             15              25     15          15
## 51                1             15              15     10          10
## 52                1             20              25     20          25
## 53                1             85              65     75          80
## 54                0             80              60     85          85
## 55                0             80              40     70          55
## 56                1             90              65     85          85
## 57                1             25              45     55           5
## 58                1             60              55     40          10
## 59                1             70              35     70          10
## 60                0             65              35     35          35
```

```
## 61                0            15              5      5          80
## 62                0            25             15     10          20
## 63                0            60             60     60          80
## 64                0            10             10     10          45
## 65                1            65             60     55          65
## 66                1            65             30     60          45
## 67                0            70             55     70          75
## 68                0            25             15     35          10
## 69                1            90             50     90          95
## 70                0            35             40     55          65
## 71                0            65             40     55          65
## 72                0            55             40     60          70
## 73                0            15              5     15           5
## 74                1            15              5     15           5
## 75                1            30             15     30          10
## 76                1            15             15     15           5
## 77                0            70             35     70          70
## 78                0            70             35     70          55
## 79                0            55             25     60          30
## 80                0            75             60     75          80
## 81                0            50             50     50          60
## 82                0            60             55     60          55
## 83                1            65             65     65          65
## 84                1            70             70     75          90
## 85                0            85             90     95         100
## 86                0            85             50     90          95
## 87                1            80             80     85          85
## 88                1            45             50     55          50
## 89                0            50             40     30          50
## 90                1            70             50     70          40
## 91                1            60             35     40          30
## 92                1            25             40     20          15
## 93                1            85             75     80          90
## 94                0            65             50     60          45
## 95                1            65             45     65          35
## 96                0            70             45     60          35
## 97                1            45             35     40          40
## 98                1            45             45     40          35
## 99                1            55             50     45          50
## 100               1            40             40     45          45
## 101               0            80             90     50          95
## 102               0            95             90     65          95
## 103               0            75             80     20          45
## 104              NA            NA             NA     NA          NA
## 105               0            30             70     15          10
## 106               0            60             25     45          15
## 107               1            35             20     20          10
## 108               1            60             70     40          60
## 109               0            70             85     90          90
## 110               0            65             55     45          30
## 111               1            35             20     30          10
## 112               0            75             60     80          70
## 113               1            NA             NA     NA          NA
## 114               1            NA             NA     NA          NA
## 115               0            NA             NA     NA          NA
## 116               1            NA             NA     NA          NA
## 117               0            NA             NA     NA          NA
## 118               0            NA             NA     NA          NA
## 119               1            NA             NA     NA          NA
## 120               0            NA             NA     NA          NA
## 121               0            15             75     20          70
## 122               0            20             20     25          15
## 123               1            40             15     35          20
## 124               1            20             15     25          10
## 125               0            85             15     95         100
## 126               0            70             45     70          80
```

```
## 127               1            20              15     25          15
## 128               0            30              35     55          45
## 129               0            30              10     15          15
## 130               1            35              10     10          15
## 131               1            40              15     20          15
## 132               1            40              20     25          20
## 133               0            80              65     80          85
## 134               0            70              60     70          70
## 135               1            65              60     65          60
## 136               1            75              65     75          75
##     sd_experience_yrs c_experience_months loc_written_pm review_months
## 1                  20                 228            200            10
## 2                  20                 228            200            10
## 3                  20                 228            200            10
## 4                  20                 228            200            10
## 5                  20                 228            200            10
## 6                  20                 228            200            10
## 7                  20                 228            200            10
## 8                  20                 228            200            10
## 9                   6                  24             20            72
## 10                  6                  24             20            72
## 11                  6                  24             20            72
## 12                  6                  24             20            72
## 13                  6                  24             20            72
## 14                  6                  24             20            72
## 15                  6                  24             20            72
## 16                  6                  24             20            72
## 17                 21                 252            100           240
## 18                 21                 252            100           240
## 19                 21                 252            100           240
## 20                 21                 252            100           240
## 21                 21                 252            100           240
## 22                 21                 252            100           240
## 23                 21                 252            100           240
## 24                 21                 252            100           240
## 25                 15                  60            500            60
## 26                 15                  60            500            60
## 27                 15                  60            500            60
## 28                 15                  60            500            60
## 29                 15                  60            500            60
## 30                 15                  60            500            60
## 31                 15                  60            500            60
## 32                 15                  60            500            60
## 33                  5                  36            200            48
## 34                  5                  36            200            48
## 35                  5                  36            200            48
## 36                  5                  36            200            48
## 37                  5                  36            200            48
## 38                  5                  36            200            48
## 39                  5                  36            200            48
## 40                  5                  36            200            48
## 41                 27                 270            200           180
## 42                 27                 270            200           180
## 43                 27                 270            200           180
## 44                 27                 270            200           180
## 45                 27                 270            200           180
## 46                 27                 270            200           180
## 47                 27                 270            200           180
## 48                 27                 270            200           180
## 49                 10                  98            200           108
## 50                 10                  98            200           108
## 51                 10                  98            200           108
## 52                 10                  98            200           108
## 53                 10                  98            200           108
## 54                 10                  98            200           108
## 55                 10                  98            200           108
```

```
## 56                 10                  98             200             108
## 57                  5                 100             600              25
## 58                  5                 100             600              25
## 59                  5                 100             600              25
## 60                  5                 100             600              25
## 61                  5                 100             600              25
## 62                  5                 100             600              25
## 63                  5                 100             600              25
## 64                  5                 100             600              25
## 65                 10                 120             100              30
## 66                 10                 120             100              30
## 67                 10                 120             100              30
## 68                 10                 120             100              30
## 69                 10                 120             100              30
## 70                 10                 120             100              30
## 71                 10                 120             100              30
## 72                 10                 120             100              30
## 73                 20                 180             500               5
## 74                 20                 180             500               5
## 75                 20                 180             500               5
## 76                 20                 180             500               5
## 77                 20                 180             500               5
## 78                 20                 180             500               5
## 79                 20                 180             500               5
## 80                 20                 180             500               5
## 81                  3                  32            1000              34
## 82                  3                  32            1000              34
## 83                  3                  32            1000              34
## 84                  3                  32            1000              34
## 85                  3                  32            1000              34
## 86                  3                  32            1000              34
## 87                  3                  32            1000              34
## 88                  3                  32            1000              34
## 89                  7                 120             500              72
## 90                  7                 120             500              72
## 91                  7                 120             500              72
## 92                  7                 120             500              72
## 93                  7                 120             500              72
## 94                  7                 120             500              72
## 95                  7                 120             500              72
## 96                  7                 120             500              72
## 97                  4                  84            3000              42
## 98                  4                  84            3000              42
## 99                  4                  84            3000              42
## 100                 4                  84            3000              42
## 101                 4                  84            3000              42
## 102                 4                  84            3000              42
## 103                 4                  84            3000              42
## 104                 4                  84            3000              42
## 105                 4                  96             100              46
## 106                 4                  96             100              46
## 107                 4                  96             100              46
## 108                 4                  96             100              46
## 109                 4                  96             100              46
## 110                 4                  96             100              46
## 111                 4                  96             100              46
## 112                 4                  96             100              46
## 113                20                 245              NA             244
## 114                20                 245              NA             244
## 115                20                 245              NA             244
## 116                20                 245              NA             244
## 117                20                 245              NA             244
## 118                20                 245              NA             244
## 119                20                 245              NA             244
## 120                20                 245              NA             244
## 121                 6                  60             300              60
```

```
## 122                6                  60            300            60
## 123                6                  60            300            60
## 124                6                  60            300            60
## 125                6                  60            300            60
## 126                6                  60            300            60
## 127                6                  60            300            60
## 128                6                  60            300            60
## 129               12                 100            100            60
## 130               12                 100            100            60
## 131               12                 100            100            60
## 132               12                 100            100            60
## 133               12                 100            100            60
## 134               12                 100            100            60
## 135               12                 100            100            60
## 136               12                 100            100            60
##     loc_reviewed_pm working_hrs            Confidence             Fitness
## 1               100           1 Somewhat not confident Not fatigued at all
## 2               100           1 Somewhat not confident Not fatigued at all
## 3               100           1 Somewhat not confident Not fatigued at all
## 4               100           1 Somewhat not confident Not fatigued at all
## 5               100           1 Somewhat not confident Not fatigued at all
## 6               100           1 Somewhat not confident Not fatigued at all
## 7               100           1 Somewhat not confident Not fatigued at all
## 8               100           1 Somewhat not confident Not fatigued at all
## 9               500           1     Somewhat confident   A little fatigued
## 10              500           1     Somewhat confident   A little fatigued
## 11              500           1     Somewhat confident   A little fatigued
## 12              500           1     Somewhat confident   A little fatigued
## 13              500           1     Somewhat confident   A little fatigued
## 14              500           1     Somewhat confident   A little fatigued
## 15              500           1     Somewhat confident   A little fatigued
## 16              500           1     Somewhat confident   A little fatigued
## 17              100           2                Neutral   A little fatigued
## 18              100           2                Neutral   A little fatigued
## 19              100           2                Neutral   A little fatigued
## 20              100           2                Neutral   A little fatigued
## 21              100           2                Neutral   A little fatigued
## 22              100           2                Neutral   A little fatigued
## 23              100           2                Neutral   A little fatigued
## 24              100           2                Neutral   A little fatigued
## 25             5000           2     Somewhat confident Moderately fatigued
## 26             5000           2     Somewhat confident Moderately fatigued
## 27             5000           2     Somewhat confident Moderately fatigued
## 28             5000           2     Somewhat confident Moderately fatigued
## 29             5000           2     Somewhat confident Moderately fatigued
## 30             5000           2     Somewhat confident Moderately fatigued
## 31             5000           2     Somewhat confident Moderately fatigued
## 32             5000           2     Somewhat confident Moderately fatigued
## 33               NA           0     Somewhat confident Moderately fatigued
## 34               NA           0     Somewhat confident Moderately fatigued
## 35               NA           0     Somewhat confident Moderately fatigued
## 36               NA           0     Somewhat confident Moderately fatigued
## 37               NA           0     Somewhat confident Moderately fatigued
## 38               NA           0     Somewhat confident Moderately fatigued
## 39               NA           0     Somewhat confident Moderately fatigued
## 40               NA           0     Somewhat confident Moderately fatigued
## 41              500           6                Neutral Not fatigued at all
## 42              500           6                Neutral Not fatigued at all
## 43              500           6                Neutral Not fatigued at all
## 44              500           6                Neutral Not fatigued at all
## 45              500           6                Neutral Not fatigued at all
## 46              500           6                Neutral Not fatigued at all
## 47              500           6                Neutral Not fatigued at all
## 48              500           6                Neutral Not fatigued at all
## 49             1000           5     Somewhat confident Moderately fatigued
## 50             1000           5     Somewhat confident Moderately fatigued
```

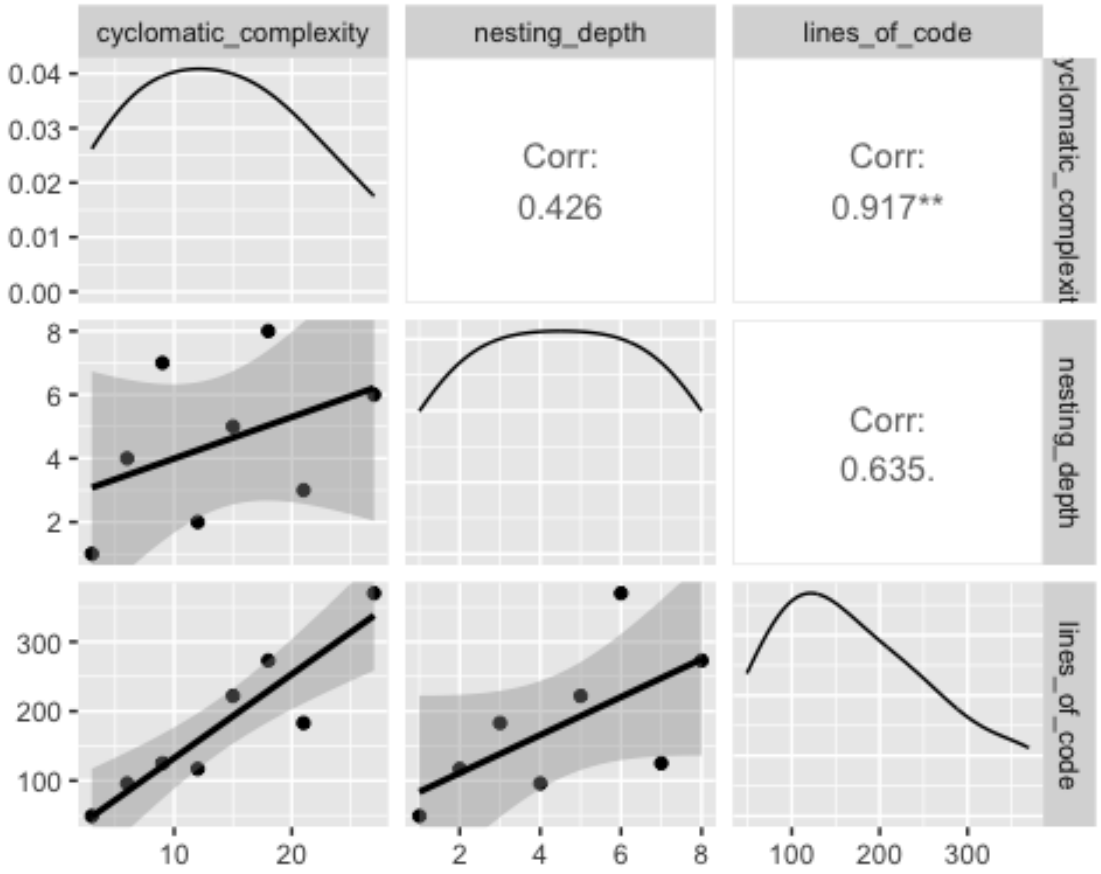
```
## 51             1000           5     Somewhat confident Moderately fatigued
## 52             1000           5     Somewhat confident Moderately fatigued
## 53             1000           5     Somewhat confident Moderately fatigued
## 54             1000           5     Somewhat confident Moderately fatigued
## 55             1000           5     Somewhat confident Moderately fatigued
## 56             1000           5     Somewhat confident Moderately fatigued
## 57               80           2     Somewhat confident Not fatigued at all
## 58               80           2     Somewhat confident Not fatigued at all
## 59               80           2     Somewhat confident Not fatigued at all
## 60               80           2     Somewhat confident Not fatigued at all
## 61               80           2     Somewhat confident Not fatigued at all
## 62               80           2     Somewhat confident Not fatigued at all
## 63               80           2     Somewhat confident Not fatigued at all
## 64               80           2     Somewhat confident Not fatigued at all
## 65              300           4    Extremely confident   A little fatigued
## 66              300           4    Extremely confident   A little fatigued
## 67              300           4    Extremely confident   A little fatigued
## 68              300           4    Extremely confident   A little fatigued
## 69              300           4    Extremely confident   A little fatigued
## 70              300           4    Extremely confident   A little fatigued
## 71              300           4    Extremely confident   A little fatigued
## 72              300           4    Extremely confident   A little fatigued
## 73              300           0     Somewhat confident Not fatigued at all
## 74              300           0     Somewhat confident Not fatigued at all
## 75              300           0     Somewhat confident Not fatigued at all
## 76              300           0     Somewhat confident Not fatigued at all
## 77              300           0     Somewhat confident Not fatigued at all
## 78              300           0     Somewhat confident Not fatigued at all
## 79              300           0     Somewhat confident Not fatigued at all
## 80              300           0     Somewhat confident Not fatigued at all
## 81             1000           4     Somewhat confident   A little fatigued
## 82             1000           4     Somewhat confident   A little fatigued
## 83             1000           4     Somewhat confident   A little fatigued
## 84             1000           4     Somewhat confident   A little fatigued
## 85             1000           4     Somewhat confident   A little fatigued
## 86             1000           4     Somewhat confident   A little fatigued
## 87             1000           4     Somewhat confident   A little fatigued
## 88             1000           4     Somewhat confident   A little fatigued
## 89              400           6     Somewhat confident Moderately fatigued
## 90              400           6     Somewhat confident Moderately fatigued
## 91              400           6     Somewhat confident Moderately fatigued
## 92              400           6     Somewhat confident Moderately fatigued
## 93              400           6     Somewhat confident Moderately fatigued
## 94              400           6     Somewhat confident Moderately fatigued
## 95              400           6     Somewhat confident Moderately fatigued
## 96              400           6     Somewhat confident Moderately fatigued
## 97             5000           1     Somewhat confident   A little fatigued
## 98             5000           1     Somewhat confident   A little fatigued
## 99             5000           1     Somewhat confident   A little fatigued
## 100            5000           1     Somewhat confident   A little fatigued
## 101            5000           1     Somewhat confident   A little fatigued
## 102            5000           1     Somewhat confident   A little fatigued
## 103            5000           1     Somewhat confident   A little fatigued
## 104            5000           1     Somewhat confident   A little fatigued
## 105              50           0     Somewhat confident Not fatigued at all
## 106              50           0     Somewhat confident Not fatigued at all
## 107              50           0     Somewhat confident Not fatigued at all
## 108              50           0     Somewhat confident Not fatigued at all
## 109              50           0     Somewhat confident Not fatigued at all
## 110              50           0     Somewhat confident Not fatigued at all
## 111              50           0     Somewhat confident Not fatigued at all
## 112              50           0     Somewhat confident Not fatigued at all
## 113              NA           0     Somewhat confident Not fatigued at all
## 114              NA           0     Somewhat confident Not fatigued at all
## 115              NA           0     Somewhat confident Not fatigued at all
## 116              NA           0     Somewhat confident Not fatigued at all
```

```
## 117              NA           0     Somewhat confident Not fatigued at all
## 118              NA           0     Somewhat confident Not fatigued at all
## 119              NA           0     Somewhat confident Not fatigued at all
## 120              NA           0     Somewhat confident Not fatigued at all
## 121              50           5     Somewhat confident   A little fatigued
## 122              50           5     Somewhat confident   A little fatigued
## 123              50           5     Somewhat confident   A little fatigued
## 124              50           5     Somewhat confident   A little fatigued
## 125              50           5     Somewhat confident   A little fatigued
## 126              50           5     Somewhat confident   A little fatigued
## 127              50           5     Somewhat confident   A little fatigued
## 128              50           5     Somewhat confident   A little fatigued
## 129             100           2     Somewhat confident Moderately fatigued
## 130             100           2     Somewhat confident Moderately fatigued
## 131             100           2     Somewhat confident Moderately fatigued
## 132             100           2     Somewhat confident Moderately fatigued
## 133             100           2     Somewhat confident Moderately fatigued
## 134             100           2     Somewhat confident Moderately fatigued
## 135             100           2     Somewhat confident Moderately fatigued
## 136             100           2     Somewhat confident Moderately fatigued

D_2 <-
  D_1 %>%
  pivot_longer(mental_demand:frustration,
               names_to = "Item",
               values_to = "response")

D_2

## # A tibble: 544 × 17
##    Part  Snippet cyclomatic_complexity nesting_depth lines_of_code   ToT
##    <chr> <chr>                   <int>         <int>         <int> <int>
##  1 4     1                           3             1            49     5
##  2 4     1                           3             1            49     5
##  3 4     1                           3             1            49     5
##  4 4     1                           3             1            49     5
##  5 4     2                           6             4            96    10
##  6 4     2                           6             4            96    10
##  7 4     2                           6             4            96    10
##  8 4     2                           6             4            96    10
##  9 4     3                           9             7           125     5
## 10 4     3                           9             7           125     5
## # ℹ 534 more rows
## # ℹ 11 more variables: Fault_detection <int>, sd_experience_yrs <int>,
## #   c_experience_months <int>, loc_written_pm <int>, review_months <int>,
## #   loc_reviewed_pm <int>, working_hrs <int>, Confidence <chr>, Fitness <chr>,
## #   Item <chr>, response <int>
```

4 Data Summary & Exploration

4.1 Metric Correlations

```
ggpairs(D_Snips, columns =2:4, lower = list(continuous = "smooth"))
```

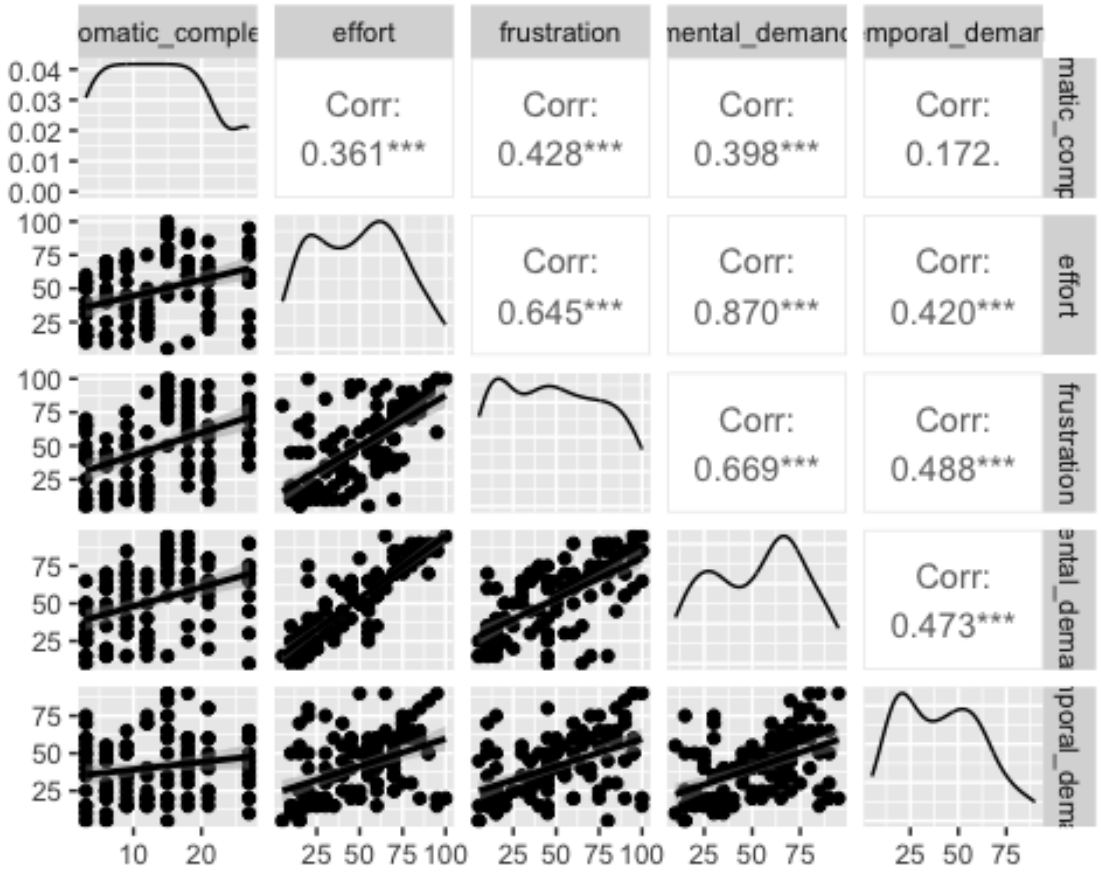


4.2 Workload

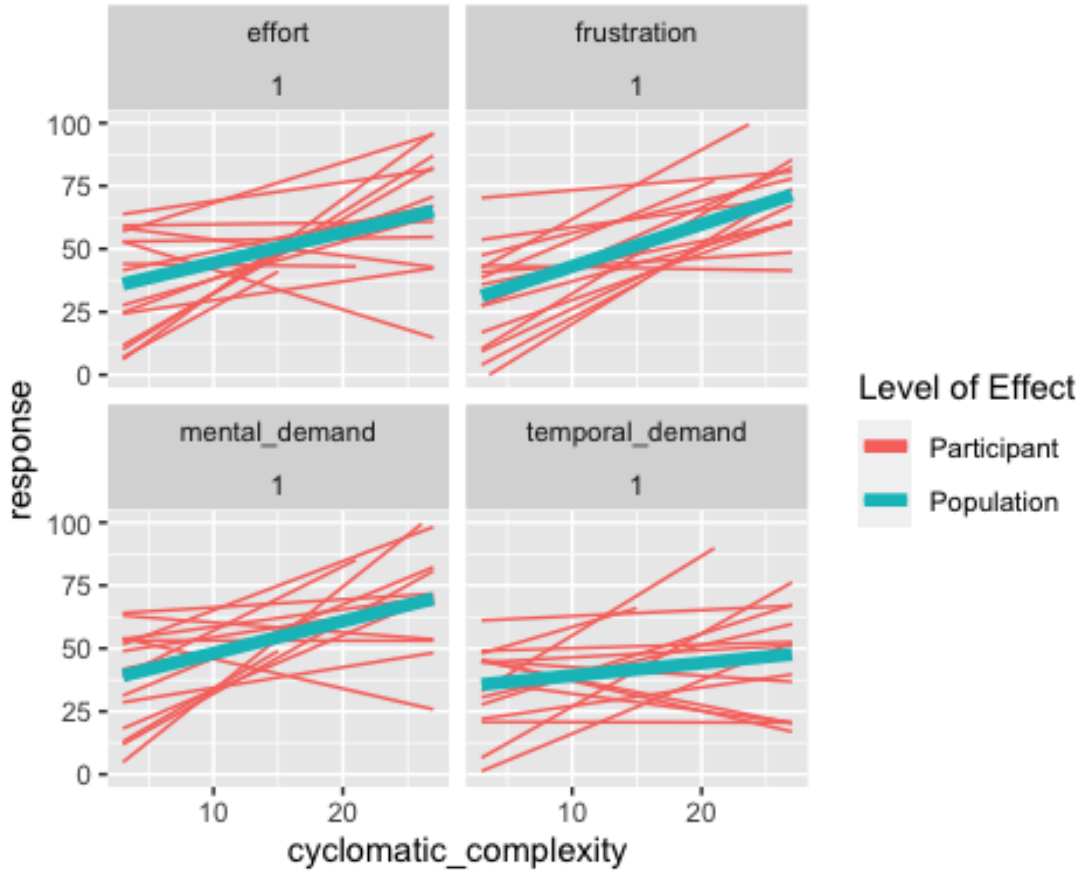
4.2.1 Cyclomatic Complexity -> Workload

```
corr_cc_workload <- D_1[, c("cyclomatic_complexity", "effort", "frustration", "mental_deman
d", "temporal_demand")]

ggpairs(corr_cc_workload, lower = list(continuous = "smooth"))

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_density()`).

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values
```

```
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_density()`).

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_density()`).

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_density()`).
```



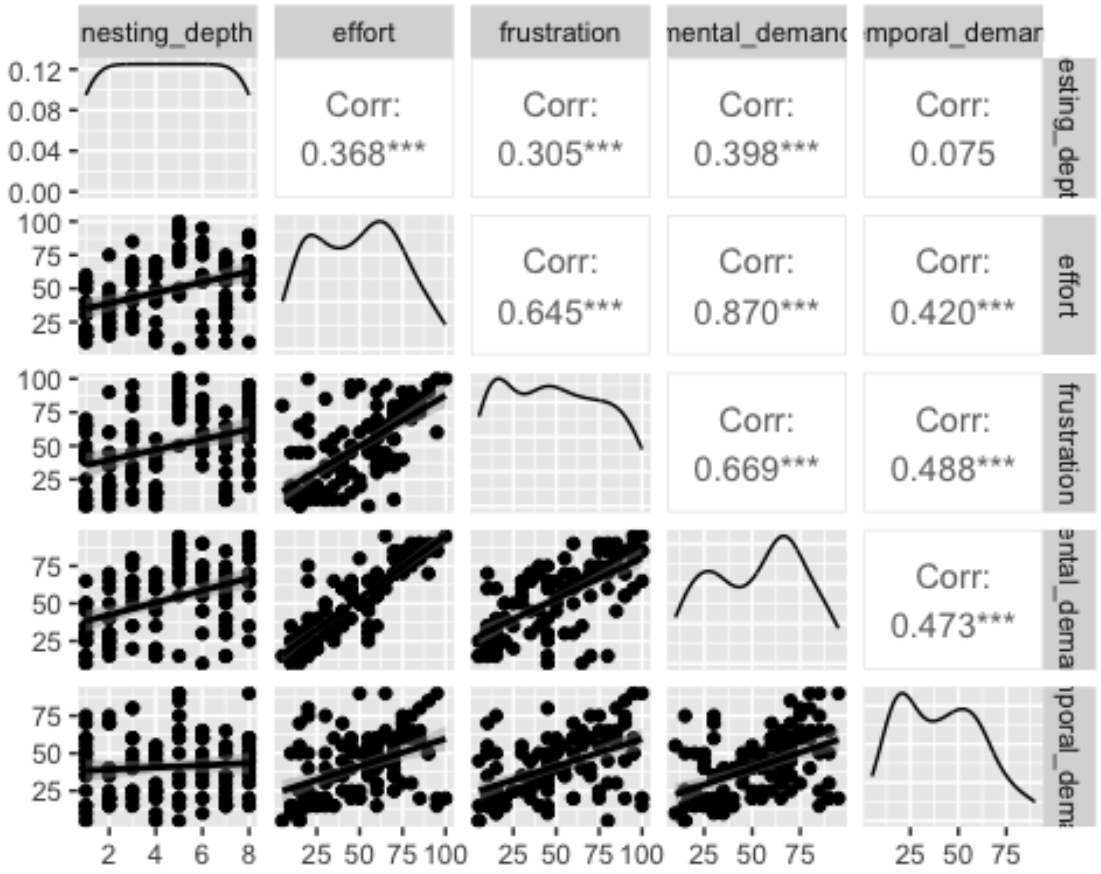

```
D_2 %>%
  ggplot(aes(x = cyclomatic_complexity, y = response, group = Part)) +
  geom_smooth(aes(colour = "Participant"), size = .5, se = F, method = "lm") +
  geom_smooth(aes(group = 1, colour = "Population"), size = 2, se = F, method = "lm") +
  labs(colour = "Level of Effect") +
  facet_wrap(Item ~ 1) +
  ylim(0, 100)

## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 84 rows containing non-finite values (`stat_smooth()`).

## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 84 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 16 rows containing missing values (`geom_smooth()`).
```



4.2.2 Nesting Depth -> Workload

```
corr_nd_workload <- D_1[, c("nesting_depth", "effort", "frustration", "mental_demand", "tem
poral_demand")]

ggpairs(corr_nd_workload, lower = list(continuous = "smooth"))

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_density()`).

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values
```

```
## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_density()`).

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_density()`).

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_density()`).
```

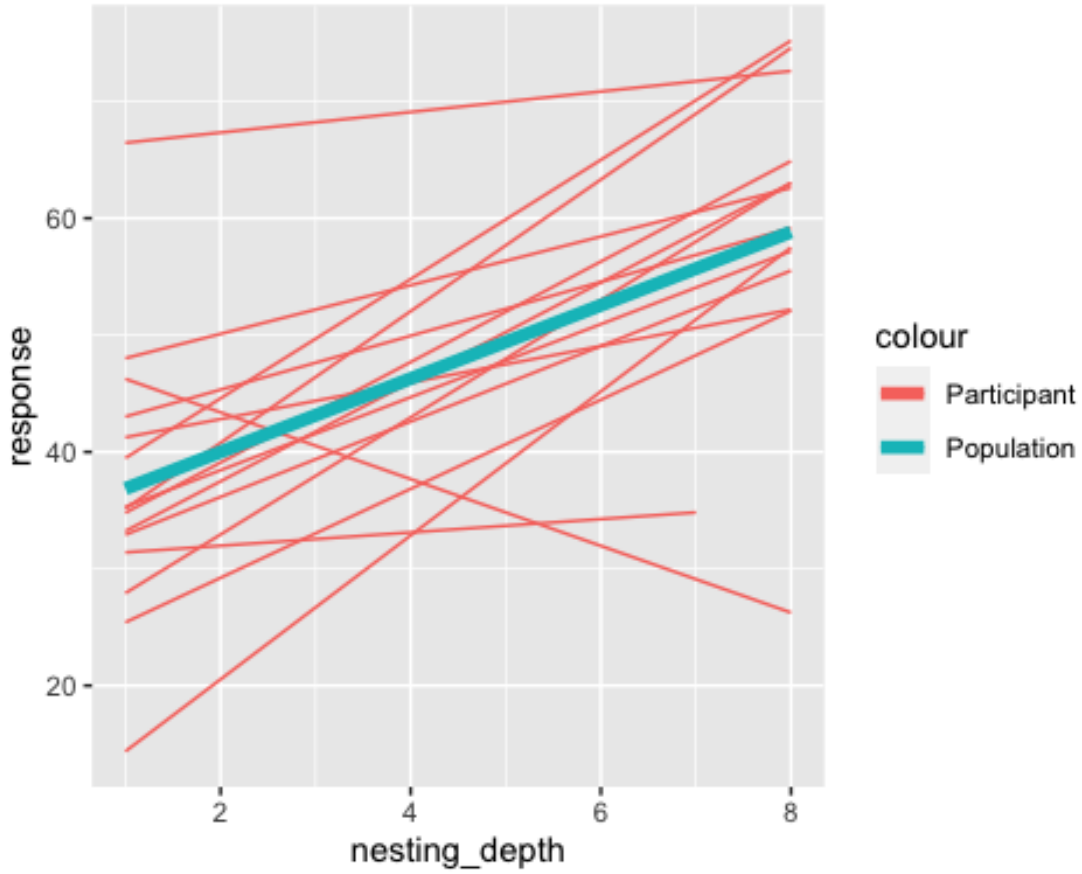



```
ggsave("scatterplot.png", plot = last_plot(), dpi = 300)

## Saving 5 x 4 in image

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_density()`).

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_density()`).

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_density()`).

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 21 rows containing missing values

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21 rows containing missing values (`geom_point()`).

## Warning: Removed 21 rows containing non-finite values (`stat_density()`).
```

```
D_2 %>%
  ggplot(aes(x = nesting_depth, y = response, group = Part)) +
  geom_smooth(aes(colour = "Participant"), size = .5, se = F, method = "lm") +
  geom_smooth(aes(group = 1, colour = "Population"), size = 2, se = F, method = "lm")
```

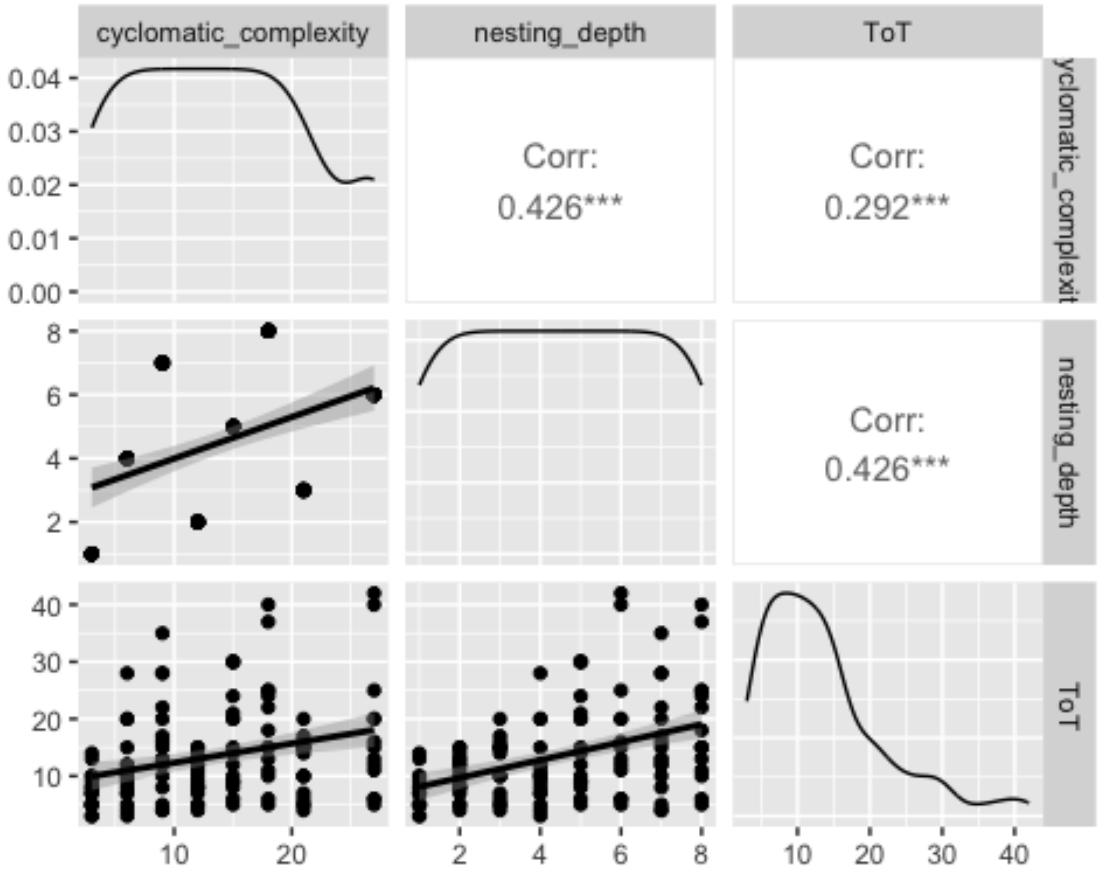
```
## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 84 rows containing non-finite values (`stat_smooth()`).

## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 84 rows containing non-finite values (`stat_smooth()`).
```




```
  labs(colour = "Level of Effect") +
  facet_wrap(Item ~ 1)

## NULL
```

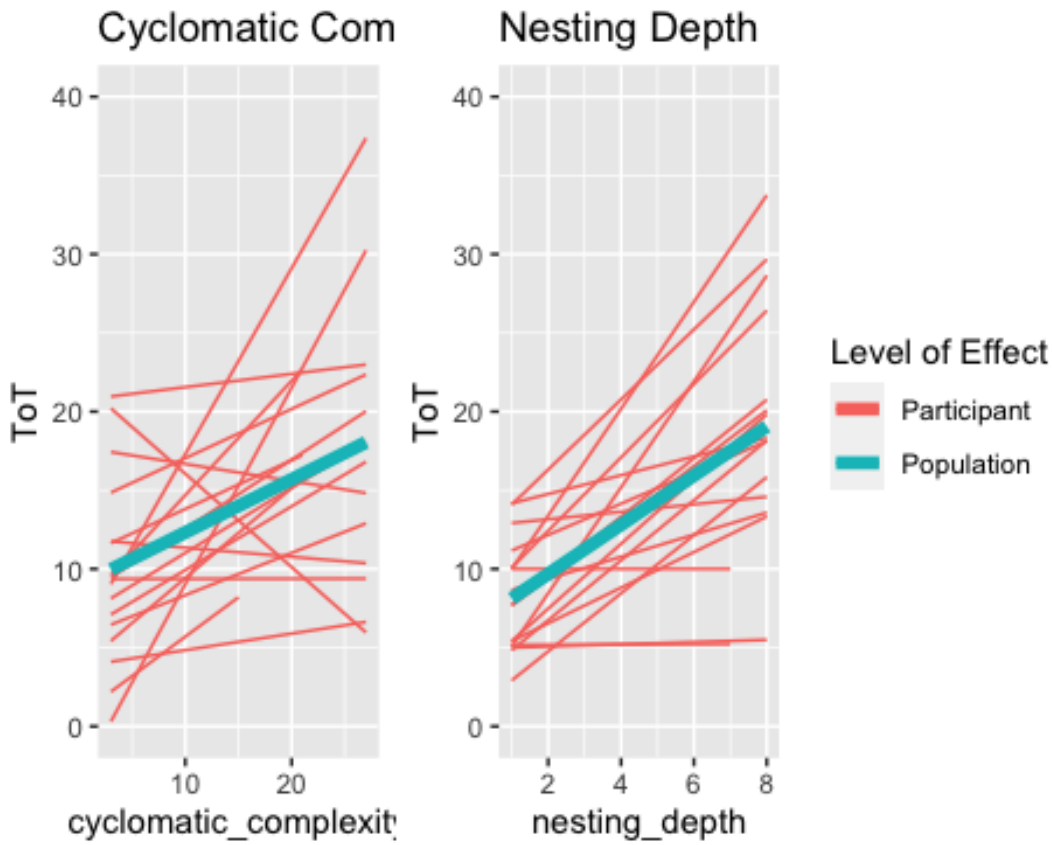
4.3 ToT

```
corr_tot <- D_1[, c("cyclomatic_complexity", "nesting_depth", "ToT")]

ggpairs(corr_tot, lower = list(continuous = "smooth"))

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 7 rows containing missing values

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 7 rows containing missing values

## Warning: Removed 7 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 7 rows containing missing values (`geom_point()`).

## Warning: Removed 7 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 7 rows containing missing values (`geom_point()`).

## Warning: Removed 7 rows containing non-finite values (`stat_density()`).
```




```
# Plot for cyclomatic complexity
plot_cc <- D_1 %>%
  ggplot(aes(x = cyclomatic_complexity, y = ToT, group = Part)) +
  geom_smooth(aes(colour = "Participant"), size = .5, se = F, method = "lm") +
  geom_smooth(aes(group = 1, colour = "Population"), size = 2, se = F, method = "lm") +
  labs(colour = "Level of Effect") +
  ggtitle("Cyclomatic Complexity") +
  coord_cartesian(ylim = c(0, 40)) +
  theme(legend.position = "none")

# Plot for nesting depth
plot_nd <- D_1 %>%
  ggplot(aes(x = nesting_depth, y = ToT, group = Part)) +
  geom_smooth(aes(colour = "Participant"), size = .5, se = F, method = "lm") +
  geom_smooth(aes(group = 1, colour = "Population"), size = 2, se = F, method = "lm") +
  labs(colour = "Level of Effect") +
  ggtitle("Nesting Depth") +
  coord_cartesian(ylim = c(0, 40))

# Combining plots
combined_plot <- plot_cc + plot_nd +
  plot_layout(ncol = 2)

print(combined_plot)

## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 7 rows containing non-finite values (`stat_smooth()`).

## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 7 rows containing non-finite values (`stat_smooth()`).
```

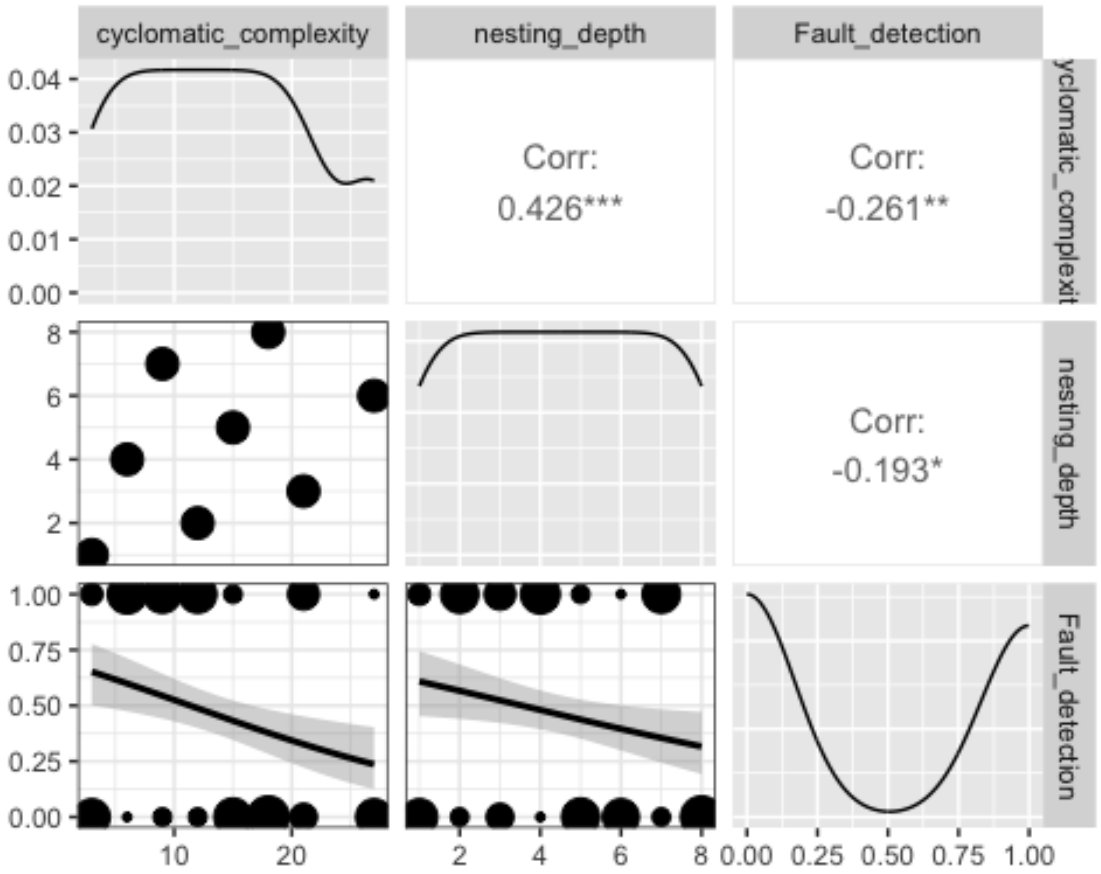
```
## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 7 rows containing non-finite values (`stat_smooth()`).

## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 7 rows containing non-finite values (`stat_smooth()`).
```



4.4 Fault Detection

```
corr_fd <- D_1[, c("cyclomatic_complexity", "nesting_depth", "Fault_detection")]

binomial_plot <- function(data, mapping, ...) {
  ggplot(data = data, mapping = mapping) +
    geom_count() +
    geom_smooth(method = "glm", method.args = list(family = "binomial"), colour = "black")
+
    theme_bw()
}

ggpairs(corr_fd, lower = list(continuous = binomial_plot))

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 4 rows containing missing values

## `geom_smooth()` using formula = 'y ~ x'

## Warning: Computation failed in `stat_smooth()`
## Removed 4 rows containing missing values
## Caused by error:
## ! y values must be 0 <= y <= 1

## Warning: Removed 4 rows containing non-finite values (`stat_sum()`).
```

```
## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 4 rows containing non-finite values (`stat_smooth()`).
## Removed 4 rows containing non-finite values (`stat_sum()`).

## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 4 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 4 rows containing non-finite values (`stat_density()`).
```



5 Model with LOC (as check)

```
bf_cont <-
  bf(mvbind(effort, frustration, mental_demand, temporal_demand, ToT) ~ cyclomatic_complexi
ty + nesting_depth + lines_of_code, family = gaussian())
bf_bi <-
  bf(Fault_detection ~ cyclomatic_complexity + nesting_depth + lines_of_code, family = bern
oulli(link = "logit"))

M_1 <- brm(bf_cont + bf_bi + set_rescor(FALSE), data = D_1, prior = set_prior("normal(0, 10
)", class = "b"))

## Warning: Rows containing NAs were excluded from the model.

## Warning: Specifying global priors for regression coefficients in multivariate
## models is deprecated. Please specify priors separately for each response
## variable.

## Compiling Stan program...

## Trying to compile a simple C file
```

```
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## clang -mmacosx-version-min=10.13 -I"/Library/Frameworks/R.framework/Resources/include" -
DNDEBUG   -I"/Library/Frameworks/R.framework/Versions/4.2/Resources/library/Rcpp/include/"
-I"/Library/Frameworks/R.framework/Versions/4.2/Resources/library/RcppEigen/include/"  -I"/
Library/Frameworks/R.framework/Versions/4.2/Resources/library/RcppEigen/include/unsupported
"  -I"/Library/Frameworks/R.framework/Versions/4.2/Resources/library/BH/include" -I"/Librar
y/Frameworks/R.framework/Versions/4.2/Resources/library/StanHeaders/include/src/"  -I"/Libr
ary/Frameworks/R.framework/Versions/4.2/Resources/library/StanHeaders/include/"  -I"/Librar
y/Frameworks/R.framework/Versions/4.2/Resources/library/RcppParallel/include/"  -I"/Library
/Frameworks/R.framework/Versions/4.2/Resources/library/rstan/include" -DEIGEN_NO_DEBUG  -DB
OOST_DISABLE_ASSERTS  -DBOOST_PENDING_INTEGER_LOG2_HPP  -DSTAN_THREADS  -DBOOST_NO_AUTO_PTR
-include '/Library/Frameworks/R.framework/Versions/4.2/Resources/library/StanHeaders/includ
e/stan/math/prim/mat/fun/Eigen.hpp'  -D_REENTRANT -DRCPP_PARALLEL_USE_TBB=1   -I/usr/local/
include   -fPIC  -Wall -g -O2  -c foo.c -o foo.o
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.2/Resources/library/Sta
nHeaders/include/stan/math/prim/mat/fun/Eigen.hpp:13:
## In file included from /Library/Frameworks/R.framework/Versions/4.2/Resources/library/Rcp
pEigen/include/Eigen/Dense:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.2/Resources/library/Rcp
pEigen/include/Eigen/Core:88:
## /Library/Frameworks/R.framework/Versions/4.2/Resources/library/RcppEigen/include/Eigen/s
rc/Core/util/Macros.h:628:1: error: unknown type name 'namespace'
## namespace Eigen {
## ^
## /Library/Frameworks/R.framework/Versions/4.2/Resources/library/RcppEigen/include/Eigen/s
rc/Core/util/Macros.h:628:16: error: expected ';' after top level declarator
## namespace Eigen {
##                ^
##                ;
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.2/Resources/library/Sta
nHeaders/include/stan/math/prim/mat/fun/Eigen.hpp:13:
## In file included from /Library/Frameworks/R.framework/Versions/4.2/Resources/library/Rcp
pEigen/include/Eigen/Dense:1:
## /Library/Frameworks/R.framework/Versions/4.2/Resources/library/RcppEigen/include/Eigen/C
ore:96:10: fatal error: 'complex' file not found
## #include <complex>
##          ^~~~~~~~
## 3 errors generated.
## make: *** [foo.o] Error 1

## Start sampling

summary(M_1)

##  Family: MV(gaussian, gaussian, gaussian, gaussian, gaussian, bernoulli)
##   Links: mu = identity; sigma = identity
##          mu = identity; sigma = identity
##          mu = identity; sigma = identity
##          mu = identity; sigma = identity
##          mu = identity; sigma = identity
##          mu = logit
## Formula: effort ~ cyclomatic_complexity + nesting_depth + lines_of_code
##          frustration ~ cyclomatic_complexity + nesting_depth + lines_of_code
##          mental_demand ~ cyclomatic_complexity + nesting_depth + lines_of_code
##          temporal_demand ~ cyclomatic_complexity + nesting_depth + lines_of_code
##          ToT ~ cyclomatic_complexity + nesting_depth + lines_of_code
##          Fault_detection ~ cyclomatic_complexity + nesting_depth + lines_of_code
##    Data: D_1 (Number of observations: 114)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Population-Level Effects:
##                                    Estimate Est.Error l-95% CI u-95% CI Rhat
## effort_Intercept                      28.06      5.31    17.29    38.51 1.00
```

```
## frustration_Intercept                   27.25      5.89    15.76    38.95 1.00
## mentaldemand_Intercept                  29.74      5.12    19.73    39.79 1.00
## temporaldemand_Intercept                35.17      5.24    24.69    45.29 1.00
## ToT_Intercept                            6.02      1.84     2.40     9.52 1.00
## Faultdetection_Intercept                 0.56      0.52    -0.43     1.59 1.00
## effort_cyclomatic_complexity            -0.48      0.84    -2.14     1.14 1.00
## effort_nesting_depth                     1.33      1.33    -1.29     3.88 1.00
## effort_lines_of_code                     0.12      0.08    -0.03     0.27 1.00
## frustration_cyclomatic_complexity       -1.70      0.91    -3.44     0.07 1.00
## frustration_nesting_depth               -1.93      1.50    -4.83     1.12 1.00
## frustration_lines_of_code                0.30      0.08     0.15     0.46 1.00
## mentaldemand_cyclomatic_complexity       0.11      0.79    -1.47     1.60 1.00
## mentaldemand_nesting_depth               1.99      1.26    -0.48     4.45 1.00
## mentaldemand_lines_of_code               0.07      0.07    -0.06     0.21 1.00
## temporaldemand_cyclomatic_complexity    -0.27      0.80    -1.84     1.29 1.00
## temporaldemand_nesting_depth            -0.99      1.30    -3.52     1.56 1.00
## temporaldemand_lines_of_code             0.08      0.07    -0.06     0.22 1.00
## ToT_cyclomatic_complexity               -0.22      0.29    -0.79     0.37 1.00
## ToT_nesting_depth                        0.94      0.47     0.02     1.86 1.00
## ToT_lines_of_code                        0.03      0.03    -0.02     0.09 1.00
## Faultdetection_cyclomatic_complexity     0.26      0.09     0.09     0.42 1.00
## Faultdetection_nesting_depth             0.34      0.15     0.05     0.64 1.00
## Faultdetection_lines_of_code            -0.03      0.01    -0.05    -0.02 1.00
##                                     Bulk_ESS Tail_ESS
## effort_Intercept                        4298     2982
## frustration_Intercept                   4322     3212
## mentaldemand_Intercept                  4328     3535
## temporaldemand_Intercept                4122     3400
## ToT_Intercept                           3477     3278
## Faultdetection_Intercept                3946     2699
## effort_cyclomatic_complexity            3474     3276
## effort_nesting_depth                    4040     3193
## effort_lines_of_code                    3405     2764
## frustration_cyclomatic_complexity       3742     3094
## frustration_nesting_depth               4354     3269
## frustration_lines_of_code               3686     3035
## mentaldemand_cyclomatic_complexity      4158     3157
## mentaldemand_nesting_depth              4564     3128
## mentaldemand_lines_of_code              3956     3323
## temporaldemand_cyclomatic_complexity    3864     2980
## temporaldemand_nesting_depth            4074     2880
## temporaldemand_lines_of_code            3612     2618
## ToT_cyclomatic_complexity               3501     2800
## ToT_nesting_depth                       4076     3021
## ToT_lines_of_code                       3362     2587
## Faultdetection_cyclomatic_complexity    3926     2942
## Faultdetection_nesting_depth            4184     3019
## Faultdetection_lines_of_code            3618     2500
##
## Family Specific Parameters:
##                     Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## sigma_effort           22.44      1.52    19.67    25.57 1.00     5742
## sigma_frustration      25.04      1.69    21.98    28.64 1.00     6393
## sigma_mentaldemand     21.31      1.45    18.68    24.39 1.00     5500
## sigma_temporaldemand   21.53      1.43    19.01    24.58 1.00     5709
## sigma_ToT               7.75      0.52     6.80     8.87 1.00     5754
##                     Tail_ESS
## sigma_effort            3031
## sigma_frustration       2699
## sigma_mentaldemand      2433
## sigma_temporaldemand    2892
## sigma_ToT               3090
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

6 Model without LOC

```
bf_cont <-
  bf(mvbind(effort, frustration, mental_demand, temporal_demand, ToT) ~ cyclomatic_complexi
ty + nesting_depth, family = gaussian())
bf_bi <-
  bf(Fault_detection ~ cyclomatic_complexity + nesting_depth, family = bernoulli(link = "lo
git"))

M_2 <- brm(bf_cont + bf_bi + set_rescor(FALSE), data = D_1, prior = set_prior("normal(0, 10
)", class = "b"))

## Warning: Rows containing NAs were excluded from the model.

## Warning: Specifying global priors for regression coefficients in multivariate
## models is deprecated. Please specify priors separately for each response
## variable.

## Compiling Stan program...

## Start sampling

summary(M_2)

##  Family: MV(gaussian, gaussian, gaussian, gaussian, gaussian, bernoulli)
##   Links: mu = identity; sigma = identity
##          mu = identity; sigma = identity
##          mu = identity; sigma = identity
##          mu = identity; sigma = identity
##          mu = identity; sigma = identity
##          mu = logit
## Formula: effort ~ cyclomatic_complexity + nesting_depth
##          frustration ~ cyclomatic_complexity + nesting_depth
##          mental_demand ~ cyclomatic_complexity + nesting_depth
##          temporal_demand ~ cyclomatic_complexity + nesting_depth
##          ToT ~ cyclomatic_complexity + nesting_depth
##          Fault_detection ~ cyclomatic_complexity + nesting_depth
##    Data: D_1 (Number of observations: 114)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Population-Level Effects:
##                                     Estimate Est.Error l-95% CI u-95% CI Rhat
## effort_Intercept                       25.71      5.30    15.33    36.30 1.00
## frustration_Intercept                  21.29      6.04     9.34    33.28 1.00
## mentaldemand_Intercept                 28.22      4.94    18.35    37.74 1.00
## temporaldemand_Intercept               33.58      4.88    23.87    42.96 1.00
## ToT_Intercept                           5.34      1.81     1.84     8.90 1.00
## Faultdetection_Intercept                1.01      0.47     0.12     1.95 1.00
## effort_cyclomatic_complexity            0.76      0.31     0.17     1.37 1.00
## effort_nesting_depth                    2.79      1.01     0.82     4.76 1.00
## frustration_cyclomatic_complexity       1.45      0.37     0.73     2.16 1.00
## frustration_nesting_depth               1.81      1.18    -0.57     4.15 1.00
## mentaldemand_cyclomatic_complexity      0.85      0.31     0.24     1.47 1.00
## mentaldemand_nesting_depth              2.83      0.96     0.99     4.76 1.00
## temporaldemand_cyclomatic_complexity    0.56      0.31    -0.06     1.18 1.00
## temporaldemand_nesting_depth            0.02      0.97    -1.91     1.92 1.00
## ToT_cyclomatic_complexity               0.14      0.11    -0.08     0.35 1.00
## ToT_nesting_depth                       1.37      0.35     0.69     2.05 1.00
## Faultdetection_cyclomatic_complexity   -0.06      0.03    -0.12    -0.00 1.00
## Faultdetection_nesting_depth           -0.07      0.09    -0.26     0.10 1.00
##                                     Bulk_ESS Tail_ESS
## effort_Intercept                       11159     2594
## frustration_Intercept                   7730     2887
## mentaldemand_Intercept                  8642     3085
```

```
## temporaldemand_Intercept                 8100     3150
## ToT_Intercept                            8600     3019
## Faultdetection_Intercept                 9196     3520
## effort_cyclomatic_complexity             6327     3157
## effort_nesting_depth                     5528     3240
## frustration_cyclomatic_complexity        5710     3522
## frustration_nesting_depth                6300     3390
## mentaldemand_cyclomatic_complexity       5645     2873
## mentaldemand_nesting_depth               5891     3294
## temporaldemand_cyclomatic_complexity     5515     3296
## temporaldemand_nesting_depth             6026     3470
## ToT_cyclomatic_complexity                6560     3470
## ToT_nesting_depth                        6855     2927
## Faultdetection_cyclomatic_complexity     5896     3577
## Faultdetection_nesting_depth             5437     3465
##
## Family Specific Parameters:
##                      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## sigma_effort            22.59      1.55    19.79    25.79 1.00     7985
## sigma_frustration       26.44      1.83    23.15    30.40 1.00     7115
## sigma_mentaldemand      21.31      1.46    18.71    24.48 1.00     6732
## sigma_temporaldemand    21.55      1.43    18.98    24.62 1.00     6573
## sigma_ToT                7.78      0.52     6.83     8.90 1.00     6725
##                      Tail_ESS
## sigma_effort             3207
## sigma_frustration        2938
## sigma_mentaldemand       3309
## sigma_temporaldemand     3272
## sigma_ToT                3159
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```