

Department of Electrical, Computer, and Software Engineering

Part IV Research Project

Final Report

Project Number: 52

Computerised Adaptive Test in Educational Assessments

Hajin Kim

Project Partner: Oscar Li

Yu-Cheng Tu (Supervisor)

Andrew Meads (Co-supervisor)

08 Oct 2022

Declaration of Originality

This report is my own unaided work and was not copied from nor written in collaboration with any other person.

Hajin

Name: Hajin Kim

ABSTRACT: Computerized Adaptive Test (CAT) is a novel testing paradigm that adjusts question difficulty according to the student's ability. CAT requires fewer items to reach an accurate ability estimation than the traditional fixed-length tests. However, implementing CAT for practical use requires an adequate understanding of CAT theory and software development knowledge, and it remains inaccessible for most developers and educators; yet, they are the primary stakeholders of CAT. Our research objective is to build a test platform that administers both the MST-CAT (Multistage test variant of CAT) and the fixed-length test and assess the MST-CAT's feasibility by comparing it to the fixed-length test. In the initial part of our research, we implemented a prototype test platform that administers both MST-CAT and fixed-length tests. The MST-CAT test path was implemented using the mstR framework, which is known as the most robust package that enables the development of MST-CAT. We used our learning from the literature on CAT and mstR documentation to adequately configure the MST-CAT platform. We have leveraged a modern web technology stack (React, Spring Boot, R, MongoDB, AWS EC2) and open-sourced our platform to make the practical implementation of MST-CAT more accessible for researchers. Subsequently, we ran a pilot test with 16 participants on our platform to compare the MST-CAT test against the fixed-length test. The post-test survey from the pilot testing shows a promising prospect for MST-CAT. There are limitations of our study, which include the size of the item bank, the sample size for our user study, and the precision of parameters used for IRT models. Overall, our study lays a robust foundation for practical implementations of MST-CAT at a small scale and explores its feasibility as an alternative to traditional fixed-length tests.

Abbreviations/Glossary

- CAT refers to Computerized Adaptive Test
- MST or MST-CAT refers to Multistage Test variant of CAT and is described in Section 2.2
- Testlet: a set of questions, usually referring to a set of questions for a single stage in MST-CAT
- TIMSS: Trends in International Mathematics and Science Study
- GRE: Graduate Record Examination
- NAPLAN: National Assessment Program - Literacy and Numeracy in Australia

Table of Contents

1.	<i>Introduction.....</i>	6
2.	<i>Literature Review.....</i>	7
2.1.	CAT administrative models	7
2.2.	General Configuration of CAT.....	7
2.2.1.	Item response model (IRT)	8
2.2.2.	Entry-level	8
2.2.3.	Scoring method (Ability estimation)	8
2.2.4.	Item pool (Item bank)	8
2.3.	Existing CAT efforts: Case Studies.....	9
2.3.1.	Case study: NAPLAN.....	9
2.3.2.	Case study: GRE.....	9
2.4.	Discussion	10
2.4.1.	Gaps in research.....	10
2.4.2.	CAT Model to use.....	10
3.	<i>Research Intent.....</i>	10
4.	<i>Test Configuration.....</i>	11
4.1.	Study Design.....	11
4.2.	Item bank.....	11
4.3.	Test Structure.....	12
5.	<i>Design.....</i>	13
5.1.	Requirements	13
5.1.1.	Test administration and persistent database.....	13
5.1.2.	Developer-friendly architecture	13
5.1.3.	User experience.....	13
5.2.	Software Architecture	14
5.2.1.	REST API	14
5.2.2.	Technological choices.....	14
5.3.	User Interface (UI) Design	15
6.	<i>Implementation</i>	15
6.1.	mstR	16
6.1.1.	Ability Estimation.....	16
6.1.2.	Testlet Selection.....	17
6.2.	Frontend Features.....	18
6.2.1.	Timer.....	18
6.2.2.	User Interface for Test Screen	18
6.2.3.	Short-Answer and Multiple-Choice questions support.....	18
6.2.4.	Edge case handling	19
6.2.5.	Premature test submission handling	19
6.3.	Backend Features.....	19
6.3.1.	Question Image Storage.....	19
6.3.2.	Question content / User management	20
7.	<i>Testing.....</i>	20
7.1.	User testing protocol.....	20
7.2.	User testing feedback.....	21
7.2.1.	Usability of the test platform	21

7.2.2.	Student's test experience.....	22
7.2.3.	Student's perception of test difficulty.....	23
7.3.	Correlation between MST-CAT score and FL score.....	23
7.4.	User Testing Limitation.....	24
7.4.1.	Small number of non-year9 test participants	24
7.4.2.	Same day test	24
7.4.3.	Test takers potentially not giving their best.....	24
8.	Future Work.....	24
9.	Conclusion.....	25
10.	Acknowledgement	25
11.	Appendix.....	26
11.1.	Appendix A: Allocation of questions for MST testlets and FL test.....	26
12.	References	27

Figures

Figure 1	Architecture of MST-CAT Platform	14
Figure 2	Test Screen Mockup (left) and Final Implementation (right)	15
Figure 3	mstR documentation on eapEst function (ability estimation)	16
Figure 4	mstR documentation on nextModule function (next module selection)	17
Figure 5	Short Answer Question (Left) and Multiple-Choice Question (Right).....	18
Figure 6	Response to 'I found MST-CAT to be less tedious [...]' by test order.....	22
Figure 7	MST-CAT vs FL score.....	23

Tables

Table 1	Content Distribution of TIMSS assessment	13
Table 2	Screen Flow for MST-CAT and FL Test Path.....	15

1. Introduction

The Computerised Adaptive Test (CAT) is a novel testing paradigm that adjusts question difficulty according to the student's ability. Its primary benefit is greater accuracy in estimating candidates' ability compared to the fixed-length tests [1]. It is a long-known issue that the fixed-length Test does not accurately evaluate a candidate's ability at high and low ability levels, as every candidate receives the same questions regardless of their ability [2]. In contrast, CAT administers questions that best fit students' abilities while iteratively adjusting its ability estimate as the student responds [3]. Hence, CAT extracts maximum information about the student regardless of their ability and requires fewer items to reach an accurate ability estimation than the fixed-length Test [1, 3, 4].

Much of the theoretical and practical basis for the Computer Adaptive Test was developed in late 1960 when Fred Lord started researching Item Response Theory (IRT) [3]. IRT is a central theory to CAT, which allows the estimation of ability independent of test item selection; it ensures that the ability estimations based on any two sets of questions are comparable [3]. Since then, researchers of Statistics, Education, and Computer Science domains have developed multiple CAT administrative models, the most notable of which are the Multistage Test (MST) and Question-Adaptive Model (QAM). Both CAT models have been used extensively in a small number of high-stakes examinations, such as the GRE and NAPLAN, as an alternative to fixed-length tests previously administered as paper tests [4, 5]. However, the current adoption rate of CAT stands low, especially in small-scale examinations. This is due to the inconvenience of implementing a reliable and accurate CAT test and the lack of studies that compare CAT and fixed-length tests directly. As it stands, fixed-length tests dominate the examination market due to their ease of implementation and reliability.

This report starts with a literature review that reviews the current efforts in CAT and their implementations. In our Test Configuration section, we outline the design of our MST-CAT and fixed-length tests. The Design and Implementation sections presents the architecture and implementation of our platform that supports administration of both fixed-length test and MST-CAT tests. Then, we discuss the results from our user study in our Testing section. Lastly, the Future Work section highlights the limitations of our study and how future implementations can be improved. Given the dearth of open-sourced implementations, we hope our prototype platform and user study will guide any researchers, teachers or developers interested in implementing MST-CAT for their needs.

2. Literature Review

The goal of CAT is to reduce test length while offering increased precision on the ability estimates. Through our survey of the CAT literature, we explore the general theory behind CAT and gather insights into how to best configure a robust and accurate model of MST-CAT.

2.1. CAT administrative models

CAT branches into two major test administrative models, which are QAM (Question Adaptive Model) and MST (Multistage Adaptive Test). These models determine how questions are administered and when the ability estimation (test score determination) occurs.

The Question Adaptive Model (QAM) determines the next question based on the ability to estimate a student's performance level. Two popular methods for item selection are Maximum Information (chooses a question that can maximise the information known about the student at a given ability estimate) and Bayesian item selection (selects a question that minimises the 'posterior belief distribution' - or uncertainty - about candidate ability estimation). Initial GRE CAT used this administrative model [4].

In the Multistage Adaptive Test (MST), a candidate does a *routing* test that is usually a small set of questions. Depending on their performance, they are assigned to different testlets (predetermined sets of items, each set classed by difficulty). Depending on the candidate's combined performance on both the first routing test and second test, they can be further routed to different tests (or MST can end here). Ability estimation takes account of all the testlets the candidates have completed. MST is popular among educational CAT implementations such as the revised GRE (Graduate Record Examination) [4] and NAPLAN (National Assessment Program - Literacy and Numeracy) [5].

2.2. General Configuration of CAT

Typical components of CAT are the item response model, item pool, entry-level, scoring method, item selection, and terminating criterion [6]. This applies to both QAM and MST variants of CAT. It is essential to consider each of these components to construct a well-designed CAT. Here, we briefly discuss each component and how they relate to effective CAT design.

2.2.1. Item response model (IRT)

The item response model maps the current ability estimate to a candidate's likely response to an item. The 4-parameter logistic (4PL) model considers three parameters: discrimination, difficulty, pseudo-guessing, and inattention. Less general variants (1PL, 2PL) are also widely used as IRT models in CAT implementation and are respectively constrained by one (discrimination) and two (discrimination, difficulty) parameters [7]. Less information on particular parameters would lead to choosing the logistic model with fewer parameters, and the 1PL logistics model is practically robust enough and widely used.

2.2.2. Entry-level

Entry-level is the initial question administered to a candidate. Entry-level is vital because it can impact the psychological state of the candidate [8]. If too difficult, the candidate might perform worse due to anxiety; if too easy, the candidate can become careless and create mistakes. Gershon [9] suggests initially administering an accessible item to minimise the effect of test anxiety on a candidate's performance. If some information about the candidate's ability is known, entry-level can consider that information for the initial item selection [10].

2.2.3. Scoring method (Ability estimation)

The ability estimation algorithm is principal to CAT, as it estimates ability at each candidate's response and ultimately provides the final estimate. Models such as ML (maximum likelihood), BM (Bayes modal), WL (weighted likelihood), and EAP (expected a posteriori) estimators are used [11], whereby ability estimate is a value that either maximises or minimises an objective function. For example, the ML estimator estimates the ability to be a value that maximises the maximum likelihood function. The details of these models are described in the paper by W. J. Linden et al. [1].

2.2.4. Item pool (Item bank)

The item bank is a collection of questions that can be administered to candidates. Estimating a candidate's ability requires enough questions of a difficulty that match the candidate's ability [10]; that is, difficult items are most informative to generate estimation about a candidate with high ability, and vice versa. Some researchers have also discovered that item parameters do not require the highest accuracy estimation: 'educated guess is good enough' [12]. It is agreed upon that experts can assign relative difficulties to items in a fairly robust manner [8].

2.3. Existing CAT efforts: Case Studies

Case studies on CAT reveal what factors are considered in designing a CAT to best serve the assessment goals. In this section, we discuss the case studies of NAPLAN and GRE.

2.3.1. Case study: NAPLAN

NAPLAN (National Assessment Program - Literacy and Numeracy in Australia) is a test that recently (in 2017) shifted from fixed-length pen and pencil test to an online CAT. When NAPLAN transitioned to CAT, CAT needed to cover the same breadth of domains, regardless of what test path a candidate takes. Consequently, ACARA (Australian Curriculum, Assessment, and Reporting Authority) chose the MST design as opposed to QAM as NAPLAN's CAT model. The main rationale for this decision was that the test-makers have a greater control over the test structure, the test administration pathways, and the exposure of items. It also allows students to review their responses and change their responses [13]. NAPLAN's MST design is 3-stage, with one starting testlet at the first stage, two testlets at the second stage, and three testlets at the third stage. In addition, NAPLAN embeds a unique feature in MST to maintain underperforming student's test engagement. When a candidate underperforms in the stage 1, they take the easiest testlet of stage 3 in the next stage, so that they are exposed to the easiest test panel early before losing motivation. Then, the student takes the easiest stage 2 testlet in the last stage. A simulation conducted to test the feasibility of this tailored design suggested that this tailored design considerably improves the measurement precision of the student achievement [13].

2.3.2. Case study: GRE

GRE (Graduate Record Examination in the United States) has been CAT since 1993 and is a famous case study for CAT. In this section, we summarise GRE's rationale for choosing MST as their test administrative model, as well as the implementation details. GRE chose QAM as its question administrative model in 1993, but they switched to MST when they redesigned GRE in 2014. QAM-based CAT does not allow test takers to review their answers once each candidate answers a question. In addition, it bears a high test development cost because of the requirement that every test path should measure each candidate equally well. MST-based CAT fixes both issues because 1) full evaluation of each testlet (or *panel* in their term) is possible before delivery (hence ensures each testlet meets various GRE test specifications), 2) limit can be imposed on testlet reuse and test item overlap between testlets thereby improving test security 3) test-takers can revisit questions before moving onto next testlet and change their answers [4]. Implementation of the current MST version of GRE consists of 2-stages, one panel at the first stage and three panels (easy, medium, difficult) at the second stage. Each testlet is 20 questions, and 2PL model is used for score estimation [4].

2.4. Discussion

2.4.1. Gaps in research

There are several gaps that we identify from our literature review. First is the lack of open-sourced MST-CAT based test platforms. Despite the presence of case studies such as GRE and NAPLAN, their implementations are close-sourced, meaning that the researchers unaffiliated with GRE/NAPLAN cannot easily access the practical implementation details for MST-CAT platforms. There is a need for an open-sourced MST-CAT test platform implementation so that there is more discussion around the practical use of MST-CAT. Second is the lack of case studies of small-scale MST-CAT implementations. Most tests take place at a classroom-scale and are not backed by resources and knowledge that enabled the large-scale implementations of GRE or NAPLAN. Last is the lack of studies that compare MST-CAT against fixed-length test in a practical setting. Not many studies have been done to evaluate the user experience of MST-CAT compared to FL, even though user acceptance is key to wider adoption of MST-CAT. We aim to fill these gaps through our study.

2.4.2. CAT Model to use

We decided to focus on implementation of MST-CAT based prototype, as there are already a few studies focusing on implementation of QAM-based test platforms, such as a prototype quiz platform by Oppl S., et al. [14] and Concerto [15]. MST-CAT has not received much attention until recently in the research and open-source domains, despite being used by GRE and NAPLAN at a large scale. A small-scale of MST-CAT based test platform prototype and a user study will meaningfully contribute to the research community. Furthermore, NAPLAN and the revised GRE both use the MST over the QAM test administrative model to benefit from lower development costs of maintaining a large pool of questions and the more robust guarantee on the fairness. Since we want to avoid large development costs, MST seems appropriate for our implementation. Given excellent recent case studies of GRE and NAPLAN on the use of MST, it would be in our best interest to explore deeper into the use of MST.

3. Research Intent

Our research intent is to design and conduct a study comparing MST-CAT and fixed-length tests. To achieve this, we aim to architect and implement a test platform that administers both MST-CAT and fixed-length tests. Then, we will conduct a user study with real participants on our platform, where the participants take both the fixed-length and the MST-CAT tests. Then, we will analyse the results from the study to evaluate the feasibility of using the MST-CAT test as an alternative to the fixed-length test.

4. Test Configuration

4.1. Study Design

Before designing our tests, it was important to establish our experimental design. To compare MST-CAT and fixed-length tests, the team had the choice of asking the test participant to take both tests (by statistical term, Within-Subject design) or take one test only (Between-Subject design) [16]. The team decided on the Within-Subject experiment design, as we can directly observe the effect of the control (test type) on the performance and experience of each participant, which is the key to comparing these two types of tests. With the Between-Subject design, we would have randomly assigned participants into two groups (one group takes the FL test, and the other group takes the MST-CAT test) to compare the experience of the two groups. However, this does not allow us to collect user feedbacks that compare MST-CAT and FL, and score comparison between these two groups will be meaningless because participants' natural performances can heavily impact group statistics. The only apparent downside of the Within-Subject experiment design is the rehearsal effect [16]. We attempted to minimise this effect by not using the same questions across MST and FL tests.

4.2. Item bank

The choice of study design naturally necessitates a large pool of question items to prevent any rehearsal effect. It is also necessary so that we can have enough questions for both the MST-CAT test and the FL, while ensuring that both tests assess a large breadth of contents and difficulty. A crucial constraint to our item bank is that each question needs to come with a difficulty assigned to it, as MST-CAT requires a difficulty parameter value.

Upon our research, we decided that a question bank from TIMSS (Trends in International Mathematics and Science Study) is appropriate for our use [17]. The released TIMSS questions come with the percentage correct from 42 countries and the international percentage correct average, which can be used as an index to the difficulty parameter of each question. Additionally, it contains a sufficient number of questions for our purpose in the breadth of domains (21 questions in Number, 30 in Algebra, 18 in Data and Chance, and 20 in Geometry). Since we want MST-CAT to cover a similar range of difficulties as FL, we assigned the questions to MST-CAT and FL alternately through the questions sorted based on the percentage correct in each domain. As the research team of this paper is based in New Zealand, we used the percentage correct of New Zealand students as a parameter to feed into the MST-CAT item bank configuration. We observed quite a few questions where the New Zealand percentage correct does not follow the international percentage correct, which indicates that replication studies will need to base their item bank preparation on the participants' nationality.

For constructing the item bank, we converted the New Zealand percentage correct to the difficulty parameter by the following equation. Note that a higher difficulty parameter should indicate that the question is more difficult.

$$difficultyParameter = 1 - percentageCorrect$$

This was the simplest conversion approach that was available to our disposal. It meets the representative condition, as the lower percentageCorrect indicates that the question is of higher difficulty, vice versa. Expert opinions are deemed sufficient for determining parameter values [8], and by extension, we consider that our method of determining the difficulty parameter is reasonable. As we can only derive the difficulty information of the TIMSS questions from the item bank, our item bank was calibrated to 1PL IRT model.

4.3. Test Structure

Every participant will be invited to take two sets of tests (fixed-length test, MST-CAT) as per the Within-Group study design. We designed MST-CAT to be two-stage MST, with 3 branching testlets (easy, medium, hard). This follows the same structure as GRE MST-CAT [4]. The initial testlet is slightly easier than the second-stage medium testlet. The design of MST-CAT was restricted by the number of released questions from TIMSS. If we introduced more stages or branches, each MST-CAT stage would have been too short to determine the candidate's ability and reliably assign the next stage testlet. The team could not use questions outside of the TIMSS item bank, as it could make the difficulty parameter inconsistent.

One key interest in this experiment is whether the *shorter* MST-CAT can sufficiently predict the score of a student with similar accuracy as the *longer* fixed-length test. Therefore, we decided that each MST-CAT path to have 20 questions in total (10 questions per testlet) and the fixed-length test to have 30 questions. Our final question allocation can be found in the **Appendix A**, with mapping of question identifiers (prefixed with M followed by a string of numbers) in TIMSS item bank to testlets. Based on the time that was given to students in TIMSS assessments, the team inferred that 1.5 minutes is sufficient for each question. We have set time proportional to the number of questions in the test, so FL test was set to 45 minutes and MST-CAT was set to 30 minutes.

We made sure that each MST-CAT testlet and FL test has the same breakdown of content domain proportion as that of TIMSS assessments (Table 1) [18]. The allocation of questions for each MST-CAT testlet and FL test can be found in **Appendix A**. We wished that we could assign more questions to each MST-CAT testlet in order to have substantial

number of questions for each content domain, but this was not possible due to the limited number of released questions from TIMSS.

Content Domains	Number	Algebra	Geometry	Data and Chance
Proportion	30%	30%	20%	20%

Table 1 Content Distribution of TIMSS assessment

5. Design

5.1. Requirements

The crucial goal of our platform is to allow researchers to experimentally compare the MST-CAT test against the fixed-length test as a baseline. The main stakeholders of this platform are researchers, developers, and students. The following requirements have been incorporated as the features of our platform.

5.1.1. Test administration and persistent database

The primary need of the researchers is to administer the MST-CAT and FL tests with ease through our platform, so that they can design an experiment to compare those two tests. They would also require features to manage questions (upload, delete, add), so that they can be flexible in what questions they examine the participants with. They would also want platform to persist the scores of each student after each test, so that they can be queried for data analysis. The researchers would also want the platform to appropriately leverage a good and robust MST-CAT framework, so that the results from the platform are reasonable and backed by an adequate theoretical basis.

5.1.2. Developer-friendly architecture

The developers would want the platform to be flexible in what frameworks they use for MST-CAT, so that they can develop the platform for their needs. To do this, the core feature (ability estimation and determination of the next MST-CAT module) should ideally be decoupled from platform design. The developers would also want the platform to provide transparency in the implementation, so that they can see how it is implemented and modify them for their needs.

5.1.3. User experience

UI of the platform to be intuitive and non-distracting. Users would expect the platform to save their progress when their machine crashes or the platform is accidentally closed, so that they can start from where they left off.

5.2. Software Architecture

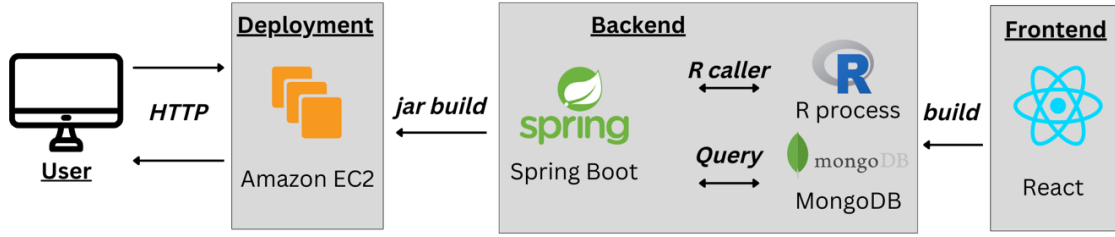


Figure 1 Architecture of MST-CAT Platform

5.2.1. REST API

Our platform follows a duo-lithic architecture where the frontend communicates to backend via HTTP – a platform agnostic communication protocol – so that platform UI and backend logic can be separately developed. Such separation makes it is easy to substitute any other frontend for our current backend. We defined REST API to support operations such as retrieving the fixed-length questions, submitting responses for a fixed-length test, retrieving MST-CAT questions for a specific module, submitting user choices for MST-CAT test, performing a CRUD operation on questions and users. Each REST API is supported by the rest controllers defined through Spring Boot backend.

5.2.2. Technological choices

For robust implementation of MST-CAT, the team used mstR R package, which is the only library that properly implements algorithms related to MST-CAT administration. mstR library supports the full configuration of MST-CAT, from IRT model parameters to estimation/scoring algorithms, making it an extremely versatile library for our implementation. We have chosen Spring Boot over alternatives such as Django/Flask (Python) and Express (Java), as only Java has a mature package that calls R functions. RCaller provides a stable and robust implementation for calling R functions from Java code [19]. It allows synchronous R code execution, which allows Java to wait until the R operation returns a result. This behavior is crucial for our project. For example, when the user submits their response for a MST-CAT testlet, the handling code needs to to block until the R process returns the result.

MongoDB was used for database as the team is most familiar with the database. Also, its flexible schema allowed us to easily add and remove fields as our understanding of necessary fields changed over time. We hosted a cloud MongoDB so that our application can remotely access the database. For deployment, we used Amazon EC2. Amazon EC2 instance

deploys the application at near-zero downtime, which ensures a smooth user experience [20]. EC2 is a Linux-based machine, allowing us to easily set up deployment environment. The R package required by RCaller is readily downloadable through yum (package-management utility in Linux).

5.3. User Interface (UI) Design

As per our requirement, the platform should administer two test paths: MST-CAT and FL tests. The team decided to determine screen flow accordingly and draw up high-fidelity mock-ups.

Steps	MST-CAT Test Path:	FL Test Path:
1	Homepage (Login screen)	Homepage (Login screen)
2	Information Screen ("You are taking first MST-CAT module")	Information Screen ("You are taking a FL test")
3	Test Screen (MST-CAT 1st stage)	Test Screen (FL test)
4	Information Screen ("Taking second module of MST-CAT")	Information Screen ("Test ended, report to the supervisor")
5	Test Screen (MST-CAT 2nd stage)	
6	Information Screen ("Test ended, report to the supervisor")	

Table 2 Screen Flow for MST-CAT and FL Test Path

Hence, the team decided to build three different pages: Homepage, Information Screen, Test Screen. Figure 2 shows the mock-up of our Test Screen and its final implementation.

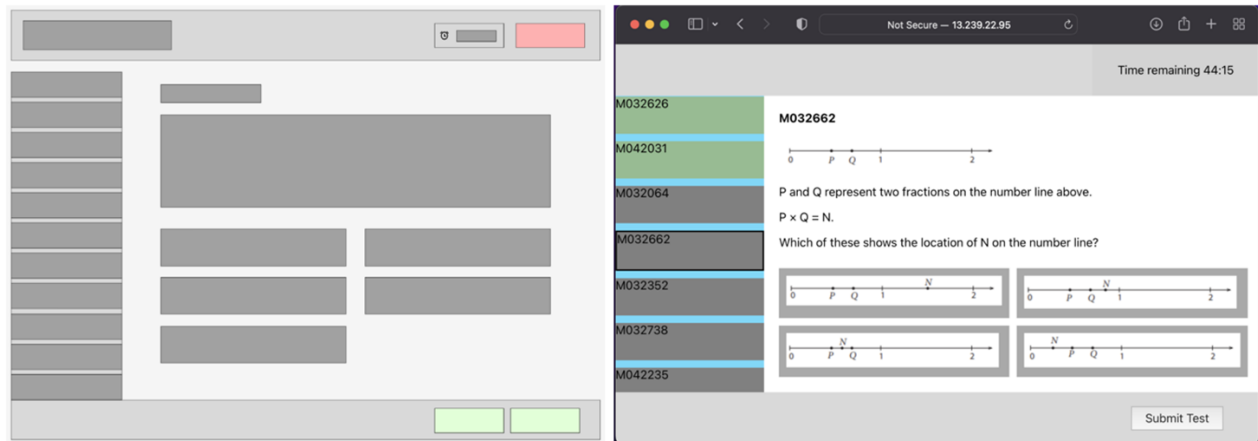


Figure 2 Test Screen Mockup (left) and Final Implementation (right)

6. Implementation

In this section, we outline how we implemented our platform and highlight the implemented features of our platform.

6.1. mstR

We leveraged algorithms implemented by the mstR framework to administer the MST-CAT test. Upon our research, we found that no study has used mstR to develop a real testing platform. All the research papers we came across so far used mstR for simulations, which limits the number of precedents that we could refer to for our development. Ultimately, we developed our codebase from scratch, relying on the official mstR documentation and learning from CAT literature. Each of these functions were stateless, meaning that we could use mstR functions with non-persistent R processes.

6.1.1. Ability Estimation

One of the mstR functions used in the platform is `eapEst`, which estimates the student's score. As shown in Figure 3, the function takes in a matrix of item parameters, item responses, model, metric constant, prior distribution, prior parameters, lower bound and upper bound for numerical integration, number of quadrature points.

<code>eapEst</code>	<i>EAP ability estimation (dichotomous and polytomous IRT models)</i>
Description	
This command returns the EAP (expected a posteriori) ability estimate for a given response pattern and a given matrix of item parameters, either under the 4PL model or any suitable polytomous IRT model.	
Usage	
<code>eapEst(it, x, model = NULL, D = 1, priorDist = "norm", priorPar = c(0, 1), lower = -4, upper = 4, nqp = 33)</code>	

Figure 3 mstR documentation on `eapEst` function (ability estimation)

For the first parameter, we provide a four-columns matrix that specifies 4 parameters for each question: difficulty, discrimination parameter, pseudo-guessing, and inattention. This is because the model supports 4PL IRT model. However, we chose 1PL model for our implementation as our information on the question is limited to difficulty parameter. We assigned default values to the other parameters, i.e., discrimination = 1, pseudo-guessing = 0, inattention = 1, which adjusts the mstR framework to operate under 1PL IRT model.

In addition, the function takes a vector of user responses to questions, where 1 indicates that the user correctly answered the question and 0 indicates the user incorrectly answered it. We assign NULL to 'model' parameter, as our IRT model is dichotomous where a response is either correct or incorrect (as opposed to polytomous, where a response can be partially correct). Other parameters such as `priorDist` and `priorPar` describe the prior knowledge on the distribution of

abilities. We assigned “norm” to priorDist, assuming that the ability distribution will follow normal distribution. Lower, upper, nqp describe parameters for lower and upper bounds of numerical integration and the number of quadrature points. Determining precise values for these parameters requires a clear internal understanding of the mstR package and the algorithms, but this is beyond the scope of this research. We assigned default values suggested from the documentation to these parameters, which we deem good enough for the purpose of our application.

6.1.2. Testlet Selection

Another function that our platform uses is nextModule function, which selects the most suitable next testlet for the student based on their performance on the previous stages. As shown in Figure 4, nextModule takes a matrix of item parameters, item membership, transition matrix, model, current module, item indicators, responses, cut-off, current ability, criterion, prior distribution, prior parameters, metric constant, ability range, quadrature points description, and random seeds.

nextModule	<i>Selection of the next module in MST</i>
Description	
This command selects the next module to be administered in the multistage test, either bases on IRT scoring or on test score and by either providing thresholds or optimally selecting the next module.	
Usage	
<pre>nextModule(itemBank, modules, transMatrix, model = NULL, current.module, out, x = NULL, cutoff = NULL, theta = 0, criterion = "MFI", priorDist = "norm", priorPar = c(0, 1), D = 1, range = c(-4, 4), parInt = c(-4, 4, 33), randomesque = 1, random.seed = NULL)</pre>	

Figure 4 mstR documentation on nextModule function (next module selection)

There are several parameters in nextModule that parallel the parameters used in eapEst. We provide the same matrix of item parameters as we do in eapEst function. However, nextModule also requires information on previously administered items. The previously administered items are recorded in the item indicators vector, where the element of that vector corresponds to the row numbers of the item bank. A response vector specifies the candidate’s responses to the items that they have received so far.

As nextModule function determines the test path, it requires multiple parameters relating to modules. We specified the modules that each item belongs to through a binary matrix, where the matrix is 1 at column j for the items that belong to module j . Also, we specify a transition matrix, which is a binary square matrix that indicates which module(s) a module can pan out to. In our application, only the first module (module 0) can pan out to other three modules. Therefore, the

transition matrix is 1 only at (0,1), (0,2), (0,3). Also, the value for current module is always 0, as we only use nextModule when the candidate transitions from the first testlet (module 0) to the next testlet. For the current ability value, we provide the output of eapEst. Cut-off matrix specifies which next module will be assigned if the current ability falls into a certain range. We have assigned NULL to it, as we decided that it will be best for mstR to determine the next module. For all the other parameters, we used the suggested default values, as we could not justify specifying any other values.

6.2. Frontend Features

6.2.1. Timer

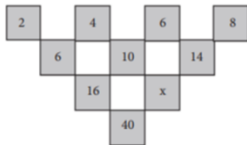
The platform features a timer. This is a must-have requirement for the test administration purpose, as MST-CAT allows students to proceed to the next stage at their desired time, and it is not possible for supervisors to manually track time left for each student. When the timer runs out, it automatically triggers test submission and directs the user to the next screen.

6.2.2. User Interface for Test Screen

We have kept UI simple for the purpose of MVP. We provide users with a clear indication of their progress by highlighting the answered questions on the side menu as green (Figure 2). Further, when an option is selected, the black border is drawn around the option box. When the user revisits the question, the option previously selected is highlighted. When the question is too large to fit on the user's screen size, the question portion of the screen becomes scrollable, allowing the user to sit the test with a variety of screen sizes.

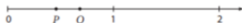
6.2.3. Short-Answer and Multiple-Choice questions support

M042228



What is the value of x in this pattern?

M032662



P and Q represent two fractions on the number line above.
 $P \times Q = N$.
 Which of these shows the location of N on the number line?

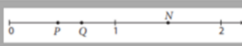
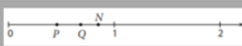
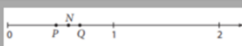
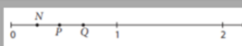





Figure 5 Short Answer Question (Left) and Multiple-Choice Question (Right)

Based on the questions from TIMSS, we found that most of the questions could be converted into a short answer or multiple-choice questions. Questions whose answer input was in a free-draw form ('Draw a tetrahedral seen from above') or free-response that can be answered in a multitude of ways ('What conclusions can you draw from seeing the histogram below?') had to be converted into multichoice questions, as smooth MST administration requires automatic marking for determining the next testlet. The platform supports multiple choice and short answer questions, and the frontend implementation can be seen in Figure 5.

6.2.4. Edge case handling

The team wanted to build a fairly robust MVP platform for our experiment. Hence, we have handled a few edge cases that could harm usability or break a user session. We protected routes except for the login page so the user cannot accidentally enter different routes when they are not logged in. Also, when the user logs in with a different id, their previous local storage will be cleared to prevent interference between different user sessions. Upon a crash or reload, the platform recovers the user's unsubmitted responses and elapsed time. Lastly, when the user submits the test with non-attempted questions, the frontend fills special values as incorrect responses to those questions so that the backend adequately processes the user responses.

6.2.5. Premature test submission handling

When the user submits the test, the platform asks for confirmation so that users are informed that they cannot revisit their answers again. This feature was implemented for better usability, as the user can accidentally submit the test.

6.3. Backend Features

6.3.1. Question Image Storage

Some of the questions from the TIMSS item bank had diagrams and charts which could not be translated into textual data. To store the images associated with the questions and their options, we explored three choices: Amazon S3, storing images as blobs directly in MongoDB, and directly storing and serving the images in the static resource folder of Spring backend. We discarded Amazon S3 solution, as scalability and security were not our biggest concern. We also discarded blob storage option, as storing images in the database is commonly unadvised except for small icons. The team decided to serve images statically from our Spring backend. We had 66 images to store and serve, and image serving latency was negligible during our user testing.

As per our requirement to build a developer-friendly architecture, our backend exposes REST API that allows the developers to manage questions and users through HTTP methods.

7. Testing

User test is critical for evaluating the usability of our platform and comparing the two different types of tests (MST-CAT, FL). We have conducted a preliminary study on 16 participants, where each participant attempted both fixed-length and MST-CAT test without minimal interruptions on the deployed platform. After the test, we gathered their feedback from the participants. The result has been anonymised and can be found in the project compendium.

The test serves the purpose of pilot testing for our prototype before it is deployed for an experiment with bigger population. All the test participants were the our friends, where 86.7% of the participants were between 19 and 22 years old, inclusive, which is beyond the target age group of TIMSS item bank. As such, the purpose of this user testing is more to gauge general feedback on the platform and test types than to assess the accuracy of testlet administration and test score estimation.

7.1. User testing protocol

The user testing largely followed the following steps.

1. At the start, the test supervisor informs the participant about the project and their tasks for the experiment.
2. We assign a numerical identifier for the test participant.
3. The test participant uses the identifier to login into the platform.
4. The test participant is randomly assigned to do MST-CAT or FL first.
5. Once the test participant completes their first test, they notify the supervisor.
6. The test participant is instructed to login again and sit the next test.
7. Once the test participant completes the second test, they fill out the post-test survey.

To minimise disruptions or technical issues during the experiment, we tested each test participant individually. The experiment time varied depending on how quickly the participant completed the tests and post-test survey, but most completed the experiment within 1 hour. Because both tests were taken back-to-back, the participant's opinions on MST-CAT and FL could be influenced by which test they took the first. For example, the participant is likely to find the second

test they took to be more tedious than the first test. To minimise this bias in the result, we have assigned the identifiers in a way that each participant receives the next number available. The test participant sat MST-CAT first if they were assigned an even number test id and FL first if odd number. This resulted in roughly 50/50 split for the test order.

7.2. User testing feedback

The following section analyses the feedback from the test participants. 16 participants attempted the tests and filled out the surveys. However, most survey questions have 15 responses, as one participant completed the survey before the survey questions were finalised. The post-test survey consists of Likert scale questions and free-response questions. All the raw data has been collated in the project compendium.

7.2.1. Usability of the test platform

Overall, the test participants report positively on the user interface and ease of use. 93.4% (14 out of 15) of the test participants found it easy to navigate the platform, 93.4% (14 out of 15) of the test participants also found the UI to be intuitive and easy to understand, and 100% (15 out of 15) of the participants found it easy to understand instructions. According to the response to the free-response question (“What did you like about the application?”), many users liked the simple design and layout, responsiveness, straightforward instruction and screen navigation. There was a positive comment on the confirmation pop-up on test submission.

However, there are lots of ways that the platform’s UI/UX can improve. On the free response question (“What did you not like about the application? (in terms of UI, features, ease of use)?”), some suggestions for improvements are: better format for displaying large diagrams, better choice of colours to make it inviting for the users, names of questions to be more informative than IDs (M032362), and better presentation of mathematical formulae (e.g., LaTeX). The responses indicate that the platform is usable, but there are rooms to improve for a better user experience.

When asked about whether the platform was suitable for testing Year 9 and 10, 86.7% (13 out of 15) responded positively. In terms of feedback on the free response question (“We plan to use this platform to test Year 9 and Year 10 students (New Zealand). Are there any changes/features that should be added to improve their testing experience?”), the suggestions were to include better graphics and implement UI that better appeals to young audiences. Some comments suggested restricted input types for short answer questions, as it can be easy to incorrectly format responses for short answer questions.

7.2.2. Student's test experience

Overall, the test participants found MST-CAT to be slightly less tedious than the FL test. When asked the Likert-scale question (“I found that Multistage test was less tedious to get through than fixed-length test”), 40% (6 out of 15) of the test participants found MST-CAT to be less tedious than the FL test, whereas 26.7% (4 out of 15) found FL test to be less tedious than MST-CAT. However, we cannot claim that MST-CAT provides a better experience than FL for different people, as the responses range from Strongly Disagree to Strongly Agree, with Neutral being the highest rated option (5 out of 15 - 33.4%), indicating either test formats can be more tedious than the other. It is interesting to note that most of those that took MST-CAT first have found MST-CAT to be more tedious than the FL test or neutral, while more than half of those that took FL first found MST-CAT to be less tedious than the FL test. This could be because the participants who have taken the FL test first are aware of the length of the FL test, and MST-CAT feels comparatively shorter.

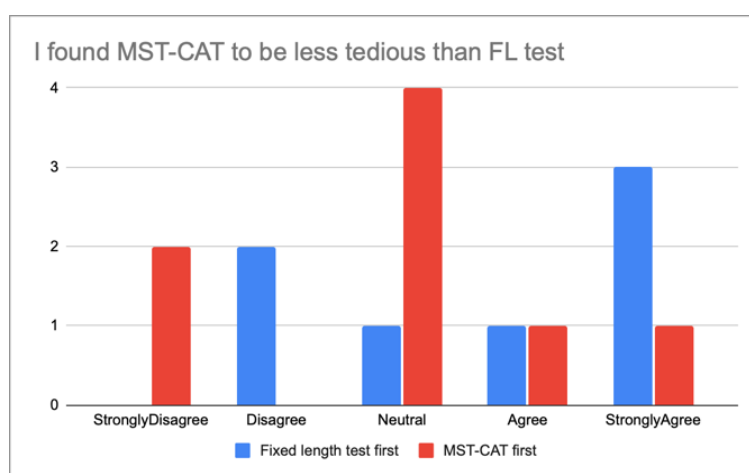


Figure 6 Response to 'I found MST-CAT to be less tedious [...]' by test order

We also asked an optional free-response survey question “Please leave any more comments or suggestions that you have. For example, what made you prefer one test over the other?”. The participants had different feelings about the format of the test. Some participants felt intimidated by 45 minutes length of the FL test than 2 x 15 minutes MST test, while others felt that they could get through the FL test quicker because there was a definitive end to the test which allowed them to know their progress. Some preferred MST purely for the lower number of questions, while others had doubt on whether MST can fully assess their ability due to its selective nature. These are very interesting feedback that shows that benefits of MST can be seen as positives and negatives for different candidates and opens room for investigation on the psychological aspect of MST-CAT.

7.2.3. Student's perception of test difficulty

Overall, the majority of the test participants found MST-CAT to better cater for their ability than the FL test. On being given a statement, “I found the Multistage Test asked questions better suited to my ability than the FL test”, 60% of the test participants agreed that MST-CAT’s difficulty was better suited to their ability than the FL test, and 26.7% were neutral about the statement. One participant reports, “the fixed length test is a bit too easy”, even though the FL test covers the same difficulty spectrum as the union of difficulty spectrum covered by all the MST testlets. This shows that MST eliminates questions that are too easy for the candidates, thereby focusing on questions that reveal the maximum information about the candidate’s ability.

7.3. Correlation between MST-CAT score and FL score

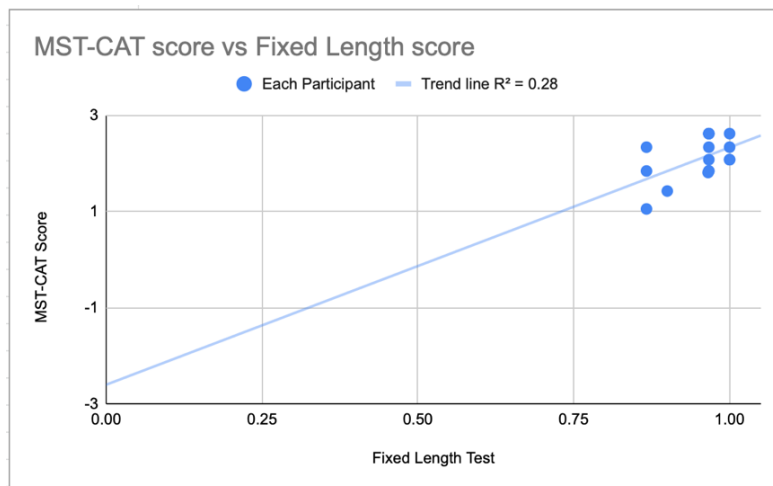


Figure 7 MST-CAT vs FL score

The graph above shows the scatter graph of each participant’s score tuple of <MST-CAT score, fixed-length Test score>. Note that fixed-length Test score’s range is [0.00, 1.00], and MST-CAT score is a random variable centred at 0 with a standard deviation of 1 (theoretically, it can take any real number). The positive correlation shows that the representative condition is met between MST-CAT and fixed-length test – that is, when the fixed-length test score is high, the MST-CAT score is high, vice versa. Assuming the fixed-length test is a reliable index of the true candidate’s ability, this shows that MST-CAT is also reflective of the true candidate’s ability. However, we have low confidence in making this claim, as the R-squared value of 0.28 shows that the correlation is weak. The spread of the MST-CAT score at each FL test score can be explained by careless mistakes.

7.4. User Testing Limitation

7.4.1. *Small number of non-year9 test participants*

Our conclusion on the effectiveness of test administration is inherently limited, as we have only tested the age-groups which are much more mathematically able than Year 9, for which this test is designed; the majority of our test participants were university students. A greater sample size would have resulted in a better correlation between MST score and FL score. If we had the chance to redo this experiment, we would have recruited a large number of Year 9 students at random in order to see if MST administers suitable testlet for students of varying mathematical abilities.

7.4.2. *Same day test*

Due to time constraint, all of our experiments were done on the same day, where the participant sits both types of test back to back without rest. There is a chance that having done FL right before MST (or converse) causes fatigue for the candidate and influence their survey results. We have tried to limit this effect by randomising the order of taking FL and MST. Spacing out the test over two days could be a good way to reduce the bias further.

7.4.3. *Test takers potentially not giving their best*

Since there is no incentive for candidates to complete the tests to the best of their ability, the scores and results seen across FL and MST may not be consistent. For example, some students might have double checked their answers in one form of test, while in other test they could have quickly finished the test. The low correlation between MST-CAT score and FL test score could be reflective of this effect. Such an effect could be mitigated with larger sample size and by recording the time spent on each test.

8. Future Work

While we tried our best to configure our prototype implementation, there are aspects that can be enhanced for a more robust implementation. Better calibration of the item bank will significantly enhance the reliability of the algorithm, as the algorithms used for ability estimation and next-module administration are directly dependent on it. The parameters used for item bank generation, score estimation, and testlet administration can be fine-tuned. Furthermore, a larger item bank will allow for a more flexible MST test design. Currently, our MST design is based on 2 stages, but this could be extended to 3 or more stages with a different number of questions. The user interface can be improved to be more suited for testing. The current prototype implements a minimal set of requirements needed to administer the questions. The

platform can be improved by adding more convenience features and custom input types for certain questions and improving the design and colour of the platform for the target audience. Finally, an ethics approval can be acquired to conduct a formal study on high school students (Year 9) using the current prototype. Through a study on high school students, we will be able to better gauge the reliability and feasibility of our prototype.

9. Conclusion

We have developed a tool that allows researchers to compare MST experimentally against the fixed-length test. The post-test survey from pilot testing shows a promising prospect for the new testing paradigm. Further fine-tuning of the platform is needed to reliably claim the feasibility of MST in various settings, but this work lays a robust baseline for future implementations.

10. Acknowledgement

With this opportunity, I would like to thank Oscar Li for jointly putting effort into this project to the end. I have always appreciated his diligence and technical aptitude. I also greatly thank Yu-Cheng Tu and Andrew Meads for their active guidance and support for our project throughout the length of this research. They have always found time to fit a weekly Zoom call in their busy lecturing and research schedule. I would also like to thank my parents and brother for their continual support. I also thank God, who is an inspiration to everything.

11. Appendix

11.1. Appendix A: Allocation of questions for MST testlets and FL test

Key: MST0 = Starting test, MST1 = Easy testlet, MST2 = Medium testlet, MST3 = Hard testlet, FL = fixed-length test

Questions of domain: Number						Questions of domain: Algebra					
QuestionID	PercentCorrectNZ	PercentCorrectInternational	IsMultichoice	Test	Difficulty	Question	% Correct NZ	% Correct International	Multichoice	Test	
M042041	79	70	FL	FL	21	M042198A	89	70	No	MST1	
M052231	70	72 No	FL	FL	30	M032295	71	73			
M032166	56	57	FL	FL	44	M032352	66	60	FL		
M052216	56	68	FL	FL	44	M052302	61	71	MST1		
M042186	50	42 No	FL	FL	50	M032738	60	65	FL		
M032626	43	49	FL	FL	57	M032797	58	54 No	MST0		
M042031	41	50	FL	FL	59	M032424	57	47	MST1		
M032064	28	27 No	FL	FL	72	M042235	57	50	FL		
M032662	19	23	FL	FL	81	M042198B	56	41 No			
M042032	58	70	MST0		42	M042236	56	58	MST0		
M032094	51	62	MST0		49	M032673	55	47	MST2		
M052061	46	41 No	MST0		54	M032477	46	46	FL		
M042024	72	54	MST1		28	M032547	43	52		Moved to Number because not enough questions	
M032595	56	49	MST1		44	M032419	42	35	MST2		
M042059	46	37 No	MST1		54	M042228	42	No	FL		
M042016	45	51	MST2		55	M042077	39	51	MST0		
M032047	43	52	MST2		57	M042066	33	32 No	MST2		
M042002	41	28 No	MST2		59	M032538	32	43 No	FL		
M052214	27	41	MST3		73	M032760A	32	27 No			
M052228	26	37	MST3		74	M042226	30	44 No	FL		
M032725	18	25 No	MST3		82	M042067	28	40	MST3		
						M032760B	25	20 No			
						M042245	20	26	FL		
						M032760C	17	15 No	MST3		
						M042198C	11	18 No			
						M050173	10	16	MST3		
						M032761	9	11 No			
						M032683	8	24 No	FL		
						M052002	5	11 No			
						M042103	2	17 No			

Questions of domain: Data and Chance					Questions of domain: Geometry				
Question	% Correct NZ	% Correct International	Multichoice	Test	QuestionID	PercentCorrectNZ	PercentCorrectInternational	Multichoice	Test
M042260	80	64		MST1	M032734	84	58 No		MST1
M042269	69	58		MST0	M032100	68	47		FL
M052429	68	45		FL	M032397	65	43		MST1
M032681A	66	60 No		MST1	M042152	63	45		MST0
M042179	66	54		MST1	M032679	57	52		MST1
M042177	63	54		FL	M042150	57	41		FL
M032681C	61	34 No		MST0	M042201	48	43 No		MST0
M032695	60	45 No		MST2	M032398	46	43		MST2
M032132	59	48		FL	M042270	46	48 No		MST2
M042207	59	47 No		MST0	M032116	45	45		FL
M042169B	56	29 No		FL	M052084	45	47		MST0
M032681B	50	29 No		MST2	M052362	40	41 No		FL
M032721	42	40		MST3	M032623	34	36		MST2
M032507	38	31		FL	M032402	32	51		
M042169A	38	43 No		MST2	M042300Z	32	31 No		no longer MST3
M042169C	24	13 No		MST3	M032324	31	39		MST3
M052503A	17	21 No		FL	M052206	27	25 No		FL
M052503B	17	17 No		MST3	M032331	26	29		MST3
					M052408	25	33 No		FL
					M032692	11	19 No		MST3

12. References

- [1] W. J. Linden, W. J. van der Linden, and C. A. Glas, *Computerized adaptive testing: Theory and practice*. Springer, 2000.
- [2] G. G. Kingsbury and D. J. Weiss, "A Validity Comparison of Adaptive and Conventional Strategies for Mastery Testing," 1981.
- [3] D. O. Segall, "Computerized adaptive testing," *Encyclopedia of social measurement*, vol. 1, pp. 429-438, 2005.
- [4] C. Wendler, B. Bridgeman, and C. Ezzo, "The Research Foundation for the GRE revised General Test: A compendium of studies," *Educational Testing Service: Princeton, NJ, USA*, 2014.
- [5] L. L. Davis, "NAPLAN ONLINE RESEARCH AND DEVELOPMENT."
- [6] R. K. Hambleton and J. N. Zaal, *Advances in educational and psychological testing: Theory and applications*. Springer Science & Business Media, 2013.
- [7] F. M. Lord, *Applications of item response theory to practical testing problems*. Routledge, 2012.
- [8] J. M. Linacre, "Computer-adaptive testing: A methodology whose time has come," MESA memorandum, 2000.
- [9] R. Gershon, "Test anxiety and item order: new concerns for item response theory," in *Objective Measurement: Theory into Practice Vol 1*.: Ablex, 1992, pp. 175-194.
- [10] D. Magis and G. Raïche, "Random generation of response patterns under computerized adaptive testing with the R package catR," *Journal of Statistical Software*, vol. 48, pp. 1-31, 2012.
- [11] D. Magis and L. Ippel, "Ability estimation (dichotomous and polytomous models)."
- [12] J. B. Bjorner, C.-H. Chang, D. Thissen, and B. B. Reeve, "Developing tailored instruments: item banking and computerized adaptive assessment," *Quality of Life Research*, vol. 16, no. 1, pp. 95-108, 2007.
- [13] L. Goran, J.-A. Justus, and S. Rabinowitz, "The Tailored test design study 2013: Summary research report," *National Assessment and Surveys Online Program, ACARA*, 2013.
- [14] S. Oppl, F. Reisinger, A. Eckmaier, and C. Helm, "A flexible online platform for computerized adaptive testing," *International Journal of Educational Technology in Higher Education*, vol. 14, no. 1, p. 2, 2017/01/20 2017, doi: 10.1186/s41239-017-0039-0.
- [15] D. Magis, "Open-source CAT software: R packages and Concerto," in *ISOQOL-NL symposium*, 2014.
- [16] G. Charness, U. Gneezy, and M. A. Kuhn, "Experimental methods: Between-subject and within-subject design," *Journal of economic behavior & organization*, vol. 81, no. 1, pp. 1-8, 2012.
- [17] T. P. I. S. Center, "Complete 2011 TIMSS 8 mathematics set," pdf 2011. [Online]. Available: https://nces.ed.gov/timss/pdf/TIMSS2011_G8_Math.pdf.
- [18] M. O. M. Ina V.S. Mullis, Graham J. Ruddock, Christine Y. O'Sullivan, Corinna Preuschoff, "TIMSS 2011 Assessment Frameworks," p. 20, 2011. [Online]. Available: https://timssandpirls.bc.edu/timss2011/downloads/TIMSS2011_Frameworks.pdf.
- [19] M. H. Satman and K. Aleksandr, "RCaller: A Java package for interfacing R," *Journal of Open Source Software*, vol. 5, no. 55, p. 2722, 2020.
- [20] P. Dalbhanjan, "Overview of deployment options on aws," *Amazon Whitepapers*, 2015.