Algorithm Lab. Journal of Selected Papers on ICAITD 2025 ICAITD 2025

Bayesian Item Response Theory and Equating Applied to Scratch Programming Courses for Upper Elementary and Junior High School Students

Ryota Kozakai† ††, Shoichiro Hara†, Yuji Watanabe†
†Nagoya City University, 1 Yamanohata, Mizuho-cho, Mizuho-ku, Nagoya 467-8501, Japan
††AKKODiS Consulting. Ltd, 3F Granpark Tower, 3-4-1 Shibaura, Minato-ku, Tokyo 108-0023, Japan
ryouta039@gmail.com, {s.hara, yuji}@nsc.nagoya-cu.ac.jp

Abstract: This study compares the analysis results for the Entry-level and Bronze-level Scratch programming courses (Kozakai, 2025) conducted from 2021 to 2023 for elementary and junior high school students at Nagoya City University. Students for the programming courses were randomly recruited from the same grade level in each course. Therefore, the population distributions of the Entry-level and Bronze-level groups were considered sufficiently similar to allow direct comparison of medians and means without performing equating. However, when tests consist of different items, equating becomes necessary, and this study provides a concrete example of such a case. Based on these results, the learning outcomes for Scratch programming from 2021 to 2023 can be confirmed.

Keywords: Programming Education, Upper Elementary School Student, Junior High School Student, Scratch, Bayesian IRT, Equating

1. INTRODUCTION

To promote information education in Japan, the new Courses of Study, which have been implemented sequentially since 2020, make “programming education” compulsory from 2020 in elementary schools, from 2021 in junior high schools, and from 2022 in high schools. It is therefore of great interest to investigate the changes that occur as the use of IT environments becomes integrated into children's daily lives. In order to investigate these changes in the environment and the current programming education for children, our laboratory has been conducting programming classes mainly for elementary, junior high, and high schools in cooperation with local schools and organizations since 2019. The Scratch language is particularly suitable for elementary and junior high school students to learn elementary programming, and Kozakai et al. (2023) held a delivery class of Scratch programming to a junior high school in Nagoya, and conducted a survey based on the results of questionnaires on interest in programming, degree of difficulty, and degree of desire to work in programming. In addition, in our previous study (Kozakai et al., 2022), we have also conducted a study on automatic grading in Python, and we are now examining whether it is possible to do the same for Scratch. There are various other educational studies based on Scratch, but they are at the level of questionnaire surveys for hands-on learning, and there are no studies that specifically examine the educational level at the current stage or show the level of understanding of each item. In this study, the estimated results for the Entry-level and Bronze-level courses are compared using horizontal equating, building upon the findings of Kozakai (2025). For the courses based on the entry-level examples, participants were upper elementary school students who attended on July 15, 2021(18 students), June 30, 2022 (19 students), and July 6, 2023(20 students). For the courses based on the bronze-level examples, participants were junior high school students who attended on November 10, 2022(20 students) and October 5, 2023(26 students). For each set of example problems, the Junior Programming Certification textbook(Fujitsu FOM Corporation, 2019) was used. We analyzed the scoring results of the program codes collected in this programming course. We analyzed the correctness and incorrectness trends of the programs entered by upper elementary and junior high school students, and investigate the difficulty and item discrimination, as well as transitions and trends in programming proficiency from year to year using Bayesian Item Response Theory. The collected data is only used for research and the secrecy is always protected. The identification of individuals will be impossible due to the masking of data.

2. RELATED WORKS

The importance of primary and secondary education for programming education has received particular attention. Kanemune(2019) describes the new curriculum that will start in the 2020 school year and the research needed in accordance with the new curriculum. In addition, Sato(2023) describes a method of supporting programming education related to Scratch. There have been studies by Hideki Mori et al. (2010, 2011) and Makoto Nakazawa et al. (2016), who investigated the practical education of Scratch and its comprehension, to analyze the reasons for not being able to understand it in their learning histories. In addition, there is a study by Susono et al. (2022) on educational

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

methods using Scratch for STEM education (Science, Technology, Engineering, and Math). While there have been many studies on new research contents, educational methods, and comprehension based on the changes in the Courses of Study, there have been few studies on learning abilities that have changed as a result of the changes, or studies on current educational issues based on specific scoring results for each code. Therefore, in this study, based on Bayesian item response theory and using the horizontal equating method, we examine changes in learning proficiency, as well as the discrimination and difficulty of the problems, based on the example problems of the entry-level and bronze-level courses. The aim of this study is to clarify the current learning ability and code-level understanding of elementary and junior high school students.

3. OVERVIEW OF THE SCRATCH PROGRAMMING LANGUAGE USED

Scratch is used in elementary education to teach what is possible through programming. Scratch can be easily assembled by simply connecting the prepared blocks, and can be used to create animations, presentations, stories, and games at any level of freedom. In Fig. 1, a crab is running away and a shark is chasing it.

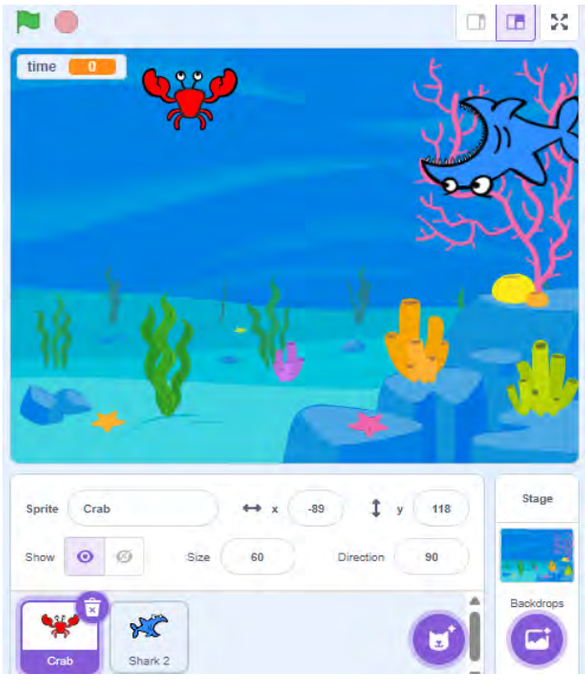


Fig 1. Diagram of a shark in pursuit and a crab fleeing.

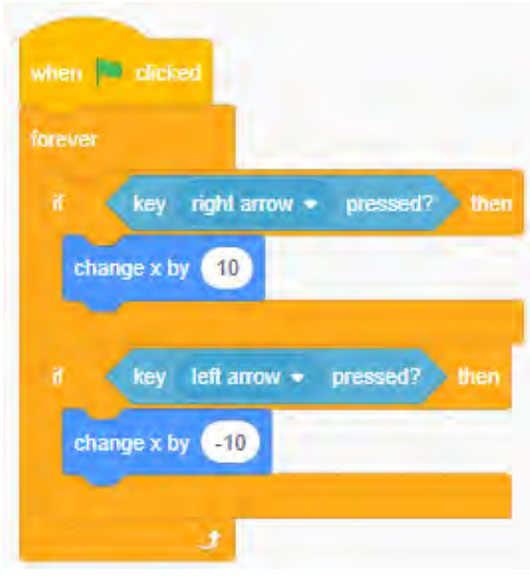


Fig 2. Program of a crab migration method.

The green flag in the upper left corner executes the program, while the pink button in the adjacent hexagon stops it. The program can be found on the 'Programming with Scratch' website ([Scratch for Beginners] Let's make a game in which you run away from sharks! (2024)) explains an example program in which the starting position of the shark is fixed at the upper right corner and the crab can move up and down, left and right. The background (Under Water is specified in Figure 1) and sprites (showing crabs and sharks) can be determined, and the size and orientation of each sprite can be specified. In addition, the behavior of each sprite can be specified by connecting blocks (Fig. 2). Fig. 2 shows the behavior of a crab created by connecting blocks. By specifying the behavior of each sprite in this way, we can specify the movement of each sprite. Many implicitly important program elements, such as conditional branching 'if ~,' repetition of 'repeat until' and 'always,' are included in the sprites. In the following sections, we will evaluate whether each block of actions is appropriately programmed by the participants based on the 'target' and 'sprite,' which indicate the target of the movement, and the 'source,' which supports the decision of the action.

4. OUTLINE OF IMPLEMENTATION OF SCRATCH PROGRAMMING COURSE

In the Scratch programming course for junior high school students in the 2021-2023 school year, exercises for the Entry level of the official textbook of the Junior Programming Proficiency Test were distributed and the course was conducted based on the contents of the textbook. The exercises consisted of programming a case that the shark is chasing fish. In the case of the fish, the program contents included the following operations.

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

1. **The display of the fish (An initial placement, when the flag is pressed):** The following actions are executed: 'Point to [90] degrees', 'x-coordinate is set to [200] and y-coordinate to [140]', 'size is set to [50] %', and 'display'.

2. **Until the fish is eaten by a shark (conditional loop):** 'Go to [mouse pointer]' is repeated until 'touched [shark]' is displayed, and 'hide' and '[eaten]' are sent when the shark touches the pointer.

3. **When the fish has escaped (conditional branch):** When the shark runs away (conditional branch): 'When [Tired] is received', 'Stop other scripts', and 'Say [I got away]' are executed in this order.

Fig. 3 shows an example of a shark, which, when the flag is pushed, performs the following actions: 'point the flag at [90] degrees', 'set the x-coordinate to [200] and the y-coordinate to [140]', 'set the size to [50%]', 'display', and when the mouse pointer is moved, the shark is hidden and 'eaten' is sent when it is touched by a shark. The program is designed to be used in a web site (Scratch). The program was created on the website (Scratch - Imagine Program Share (2019)). The program does not require any downloading, and can be used on the web screen. The program contents of the shark are as follows.

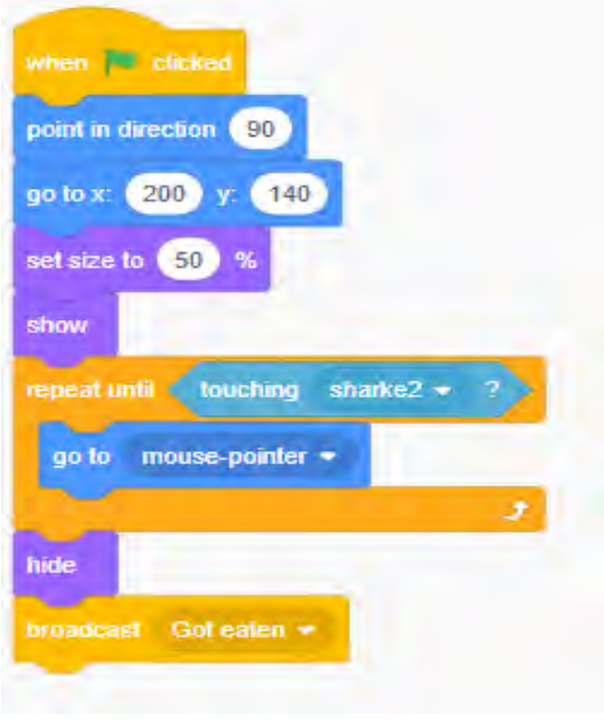


Fig. 3. Example of the program when the fish is touched from the display of the shark (when the flag is pressed).

1. **Placement and display of the shark (when the flag is pressed):** the following actions are performed, 'Rotate freely', 'x-coordinate is set to [-200], y-coordinate is set to [-150]', and 'size is set to [80] %'.

2. **Tracking of sharks to the fish (finite loop):** The following actions are performed in this order: 'point to [fish]', 'move [5] steps' is repeated 100 times, then 'stop other scripts in the script', 'set the costume to [shark-c]', 'say [tired]', and 'send [tired]' is sent. 3.

3. **When a shark eats the fish (conditional branch):** In this order, 'when [eaten] is received', 'stop other scripts in the script', and 'say [tasty]' are executed in this order.

4. **The opening and closing of the mouth of the shark when it chases the fish (infinite loop, when the flag is pressed):** The shark continuously opens and closes its mouth when it chases a shark. In order to reproduce this motion, the loop of 'always' repeats 'set costume to [shark-a]', '[0.1] seconds wait', 'set costume to [shark-b]', and '[0.1] seconds wait' in sequence.

The program is shown in Fig. 4. The program shown in Fig. 4 is an example of a shark. When the flag is pressed, the following actions are performed: 'set the rotation method to [free rotation]', 'set the x-coordinate to [-200] and the y-coordinate to [-150]', 'set the size to [80]%', 'move [5] steps' is repeated 100 times from 'point to [fish]', and then 'say [tired]' to 'send [tired]' in that order.

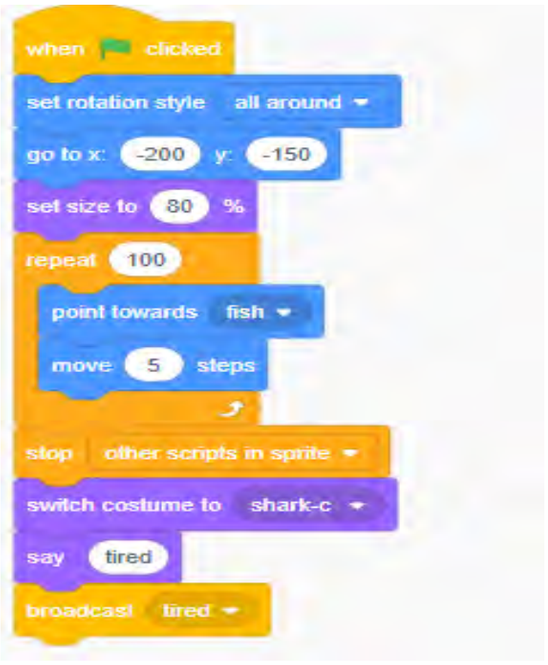


Fig. 4. An example program of a shark chasing the fish(when the flag is pressed).

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

The input program files were collected in the form of sb files. The data were collected with the full approval of the participants. For the Bronze level, the test items are as follows. The program consists of three components: the wizard, the dragons, and the spells launched by the wizard. The behaviors of these three components are described below. We begin with the program for the wizard.

1. **The display of the wizard and set an initial state(An initial placement, when the flag is pressed):** The following actions are executed: 'Set volume to [50]%', 'Point in direction [90]', 'Set size to [70]%', 'x-coordinate is set to [-180] and y-coordinate to [-100]' and 'display'. Then, 'Set [score] to [0]', and 'Say [Start!] for [1] seconds'.

2. **When the wizard moves (conditional branch):**'If the [Up Arrow] key is pressed then change the y-coordinate by [5] each time', 'If the [Down Arrow] key is pressed then change the y-coordinate by [-5] each time'.

3. **When the dragon touches the wizard (conditional branch):** 'Play sound [screech]', 'Say [I'm defeated] for [2] seconds', and 'Stop [all]' are executed in this order.

The program contents of the spell are as follows.

1. **The display of the spell and set an initial state (An initial placement, when the flag is pressed):** The following actions are executed: 'Set volume to [50]%', 'Point in direction [90]', 'Set size to [50]%', 'Hide the spell'.

2. **When a spell is fired (conditional branch):** Pressing the space bar launches the spell, plays the sound [laser2], and makes the spell travel from the wizard to the edge of the screen.

The program contents of the dragon are as follows.

1. **The display of the dragon and set an initial state (initial placement, when the flag is pressed):** The following actions are executed: 'Set volume to [30]%', 'Point in direction [-90]', 'Set size to [80]%', 'Hide the dragon'.

2. **How to show the dragon(infinite loop, when the flag is pressed):** Set costume to [dragon1-a]. Set x-coordinate to [160] and y-coordinate to pick random –120 to 120. To space out the dragon's appearances, wait pick random 1 to 3 seconds, then create the dragon.

3. **Dragon behavior (infinite loop, conditional branch, when the flag is pressed):** The dragon moves from its random starting position toward the wizard's side of the screen. Delete the dragon when it touches a spell or when it reaches the edge. If it touches a spell, add 1 to the score and play the hit sound.

5. PARTICIPANTS AND DATA COLLECTION

Participants in this course were recruited from upper-grade elementary school students (Entry-level) and junior high school students (Bronze-level) residing in Nagoya City, and were randomly selected from among the applicants. Since participation was limited to children who expressed interest in programming at the time of application, the population's level of interest in programming is considered relatively homogeneous. In addition, because simple random selection was conducted through a lottery, systematic selection bias is unlikely to have occurred. Furthermore, the grade composition of participants consisted solely of upper-grade elementary and junior high school students, and the variation in age within each group was limited. Taken together, the participant groups in this study are considered to broadly reflect the characteristics of the applicant population.

6. ON THE UNIDIMENSIONALITY OF THE ITEM RESPONSE THEORY

In this chapter, we discuss unidimensionality (Kato et al. (2014)) in item response theory. In order to show that only a single θ representing Scratch programming ability is included in the present study, we obtained a categorical correlation matrix and compared the results in terms of the magnitude of its eigenvalues. Figures 5, 6, 7, 8, 9 show very high values for the first eigenvalue. This means that unidimensionality is confirmed.

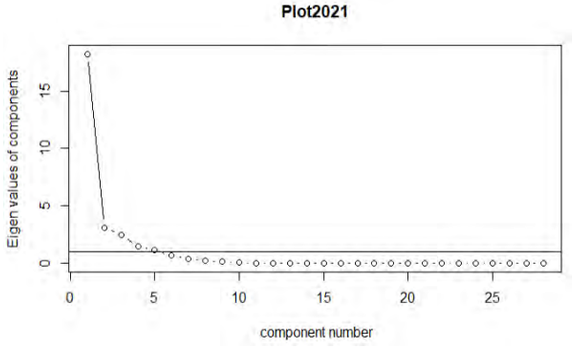


Fig. 5. The scree plot of the eigenvalues in 2021(Entry-level test)

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

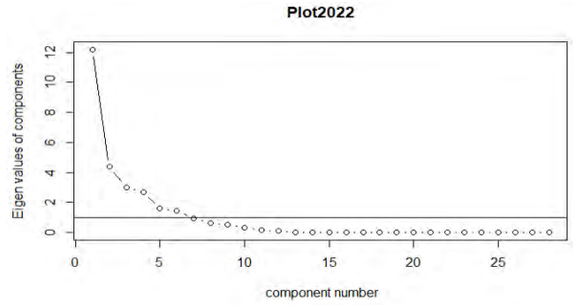


Fig. 6. The scree plot of the eigenvalues in 2022(Entry-level test)

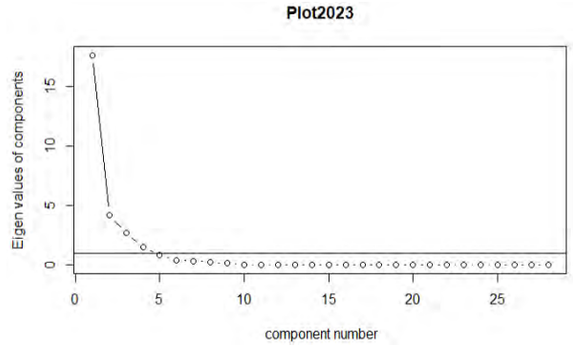


Fig. 7. The scree plot of the eigenvalues in 2023(Entry-level test)

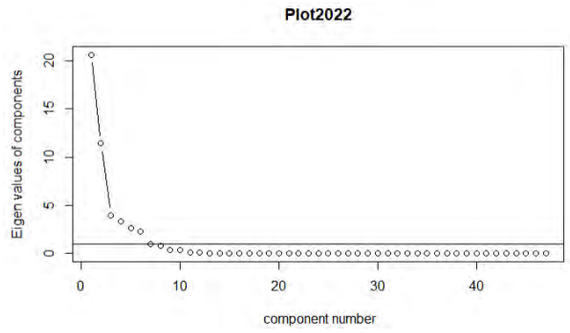


Fig. 8. The scree plot of the eigenvalues in 2022(Bronze-level test)

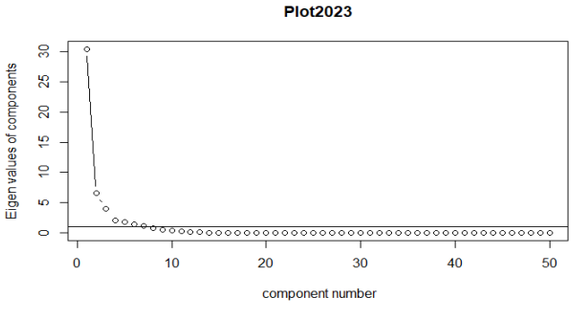


Fig. 9. The scree plot of the eigenvalues in 2023(Bronze-level test)

7. ANALYSIS OF LEARNING PROFICIENCY, DIFFICULTY, AND DISCRIMINATION

Based on the scoring results of the subjects' codes collected from 2021 to 2023, we analyzed students' proficiency levels, item difficulty, and item discrimination using Bayesian Item Response Theory. Bayesian Item Response Theory has long been used as a fair assessment method for tests (Kato et al. (2014)) and has been used for English TOEIC and mathematics questions (Ozaki et al. (2007)) (Tsukihara et al. (2008)). The likelihood of the model (N subjects in total, n items) is

$$\prod_{i=1}^{N}\prod_{j=1}^{n} P_j(\theta_i, a_j, b_j)^{\delta_{ij}}(1 - P_j(\theta_i, a_j, b_j))^{1-\delta_{ij}}$$

where the probability of correct answers is

$$P_j(\theta_i, a_j, b_j) = \frac{1}{1 + e^{-1.7a_j(\theta_i - b_j)}}$$

, θ_i is the learning proficiency level, a_j is the item discrimination, and b_j is the item difficulty level. Let $\delta_{ij} = 1$ if subject i correctly solved item j, and $\delta_{ij} = 0$ otherwise. For example, when $\theta_i = 0$, the probability of correct answer $P_j(\theta_i, a_j, b_j)$ is determined only by the item discrimination and item difficulty. If $b_j = 0$, it is determined only by the item discrimination and learning proficiency. When a_j is higher than 1, it tends to significantly increase the percentage of correct answers by participants with high learning proficiency, and vice versa. Possible cases are as follows

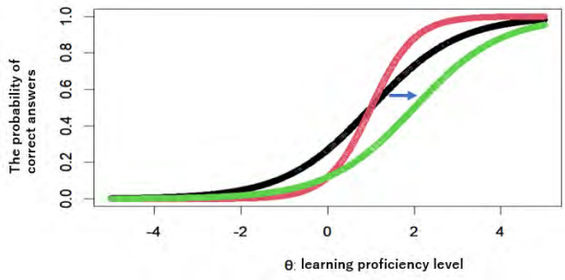


Fig. 10. Graph of curve in probability of correct answer(Black : $a_j = b_j = 1$, Red : $a_j = 2, b_j = 1$, Green : $b_j = 2, a_j = 1$).

I. The item provides a certain degree of mastery, but it is difficult to obtain a high degree of mastery for the ability to learn (low discrimination, low difficulty).

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

II. difficult as the item but good for participants with high proficiency (high discrimination and high difficulty)

III. inappropriate as the item (low discrimination, high difficulty)

IV. appropriate as the item and relatively easy (high discrimination, low difficulty)

Bayesian modelling is performed on this model. This method allows analysis on small samples. Therefore, we developed the model for Bayesian estimation. The model structure and the estimation algorithm are described below. The posterior distribution is set to

$$\prod_{i,j}\left(P(a_j,b_j,\theta_i)\right)^{\delta_{ij}}\left(1-P(a_j,b_j,\theta_i)\right)^{1-\delta_{ij}} \times \pi(a_j)\pi(b_j)\pi(\theta_i)$$

where

$\pi(a_j)\sim N(1,1)$, $\pi(b_j)\sim N(0,1)$, $\pi(\theta_i)\sim N(0,1)$. The Metropolis-Hastings method was used as the computational algorithm for the analysis. In the Metropolis-Hastings method, for a vector ϑ_k of parameters, the posterior sample is ϑ_k^j(where j is the number of updates and k is the number of parameter clusters). Then, introducing the proposed distribution $p(\vartheta_k^{j+1},\vartheta_k^j)$, sampling is performed and the acceptance probability is

$$\min\left(1,\ \frac{\frac{f(X_n|\vartheta_k^{j+1})\pi(\vartheta_k^{j+1})}{p(\vartheta_k^{j+1},\vartheta_k^j)}}{\frac{f(X_n|\vartheta_k^j)\pi(\vartheta_k^j)}{p(\vartheta_k^j,\vartheta_k^{j+1})}}\right)$$

and ϑ_k^{j+1} is adopted according to this probability. Otherwise, ϑ_k^j remains. Therefore, the proposal probabilities are as follows

$$p(a_j^{K+1},a_j^K) \text{ is } a_j^{K+1}\sim N(a_j^K,1),$$
$$p(b_j^{K+1},b_j^K) \text{ is } b_j^{K+1}\sim N(b_j^K,1),$$
$$p(\theta_i^{K+1},\theta_i^K) \text{ is } \theta_i^{K+1}\sim N(\theta_i^K,1)$$

($p(a_j^K,a_j^{K+1})$ as well.). The initial values of a_j^0, b_j^0, θ_i^0 are uniformly set to 2, 2, and 1, respectively. When the standard MH algorithm (γ = 1) is applied to imbalanced response data, posterior sampling tends to converge toward the posterior mean—a compromise solution biased toward the majority class (correct responses)—leading to underestimation of the discrimination parameter a_j. Since MAP estimation, which adopts the mode of the posterior distribution, theoretically yields a higher a_j and improved specificity for such imbalanced data, we adopt the update rule $\vartheta_k^j \to (1-\gamma)\vartheta_k^j + \gamma\vartheta_k^{j+1}$ with γ = 0.01. This scheme can be interpreted within the stochastic approximation framework of Robbins and Monro (1951), providing convergence to the neighborhood of the MAP solution after 20,000 iterations, as confirmed by the simulation results presented in the Appendix. In light of these properties, the proposed method is characterized as ***"Kozakai's gradient-free stochastic smoothing optimization"***, which inherits the proposal mechanism of MH while directing updates toward the MAP solution via smoothed stochastic steps. To evaluate the stability of the iterative procedure, the algorithm was repeated about 5 times with identical initial values. After 20000 iterations, nearly identical log-posterior values were obtained across runs (Figures 11~15). The AUC and the confusion matrix (Table 1,2) are calculated with the probability of 0.5 or higher as the correct answer state of the scoring data, and the other values as the incorrect answers. The ROC curves are shown below (Figures 16~20).

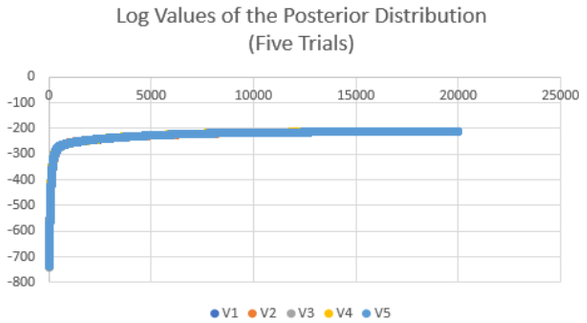


Figure 11. Convergence of Log-Posterior Values During Iterative Computation (2021 Entry-Level Course)

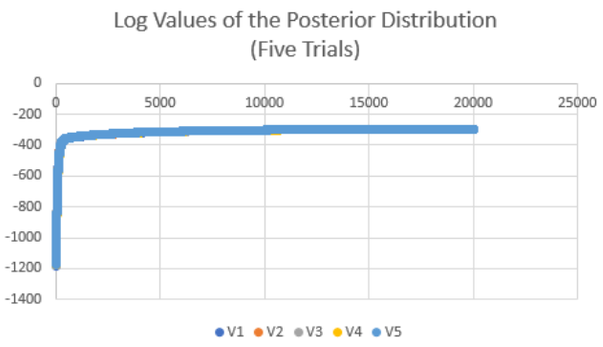


Figure 12. Convergence of Log-Posterior Values During Iterative Computation(2022 Entry-Level Course)

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

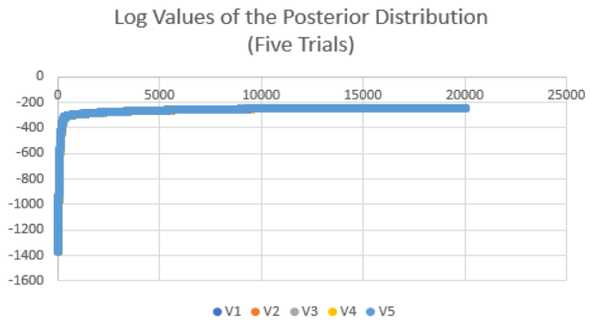


Figure 13. Convergence of Log-Posterior Values During Iterative Computation(2023 Entry-Level Course)

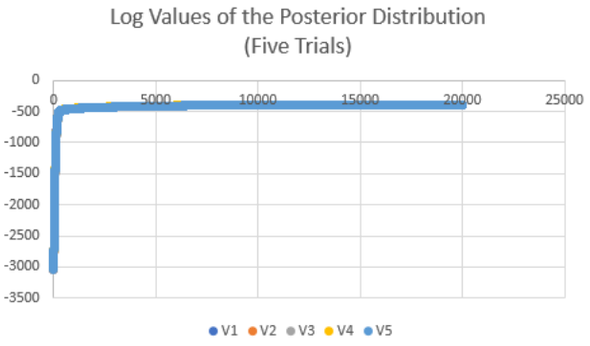


Figure 14. Convergence of Log-Posterior Values During Iterative Computation(2022 Bronze-Level Course)

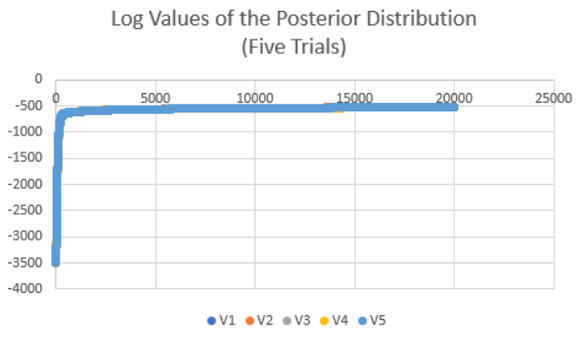


Figure 15. Convergence of Log-Posterior Values During Iterative Computation(2023 Bronze-Level Course)

Table 1. Calculation results of the confusion matrices for FY2021/2022/2023(Entry Level).

FY2021/ 2022/2023	Correct answer (Predict)	Incorrect answer (Predict)
Correct answer (observed)	147/262/328	24/39/30
Incorrect answer (observed)	21/66/31	312/165/171

Table 2. Calculation results of the confusion matrices for FY2022/2023(Bronze Level).

FY 2022/ 2023	Correct answer (Predict)	Incorrect answer (Predict)
Correct answer (observed)	789/910	30/42
Incorrect answer (observed)	81/119	100/229

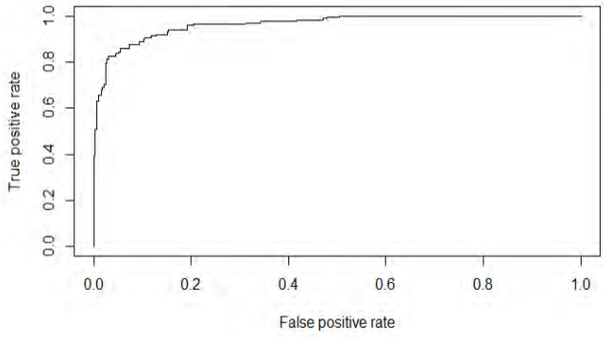


Figure 16. Diagram of ROC curve for FY2021 (Entry Level, AUC:0.96).

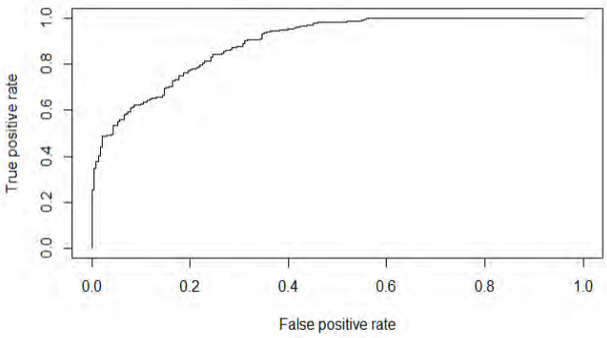


Figure 17. Diagram of ROC curve for FY2022 (Entry Level, AUC:0.89).

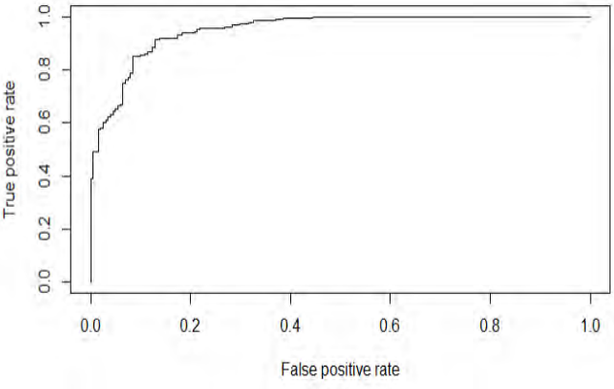


Figure 18. Diagram of ROC curve for FY2023 (Entry Level, AUC:0.95).

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

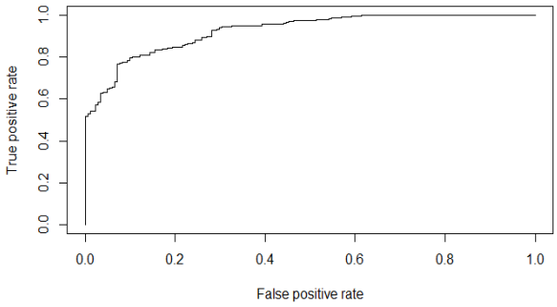


Figure 19. Diagram of ROC curve for FY2022 (Bronze Level, AUC:0.92).

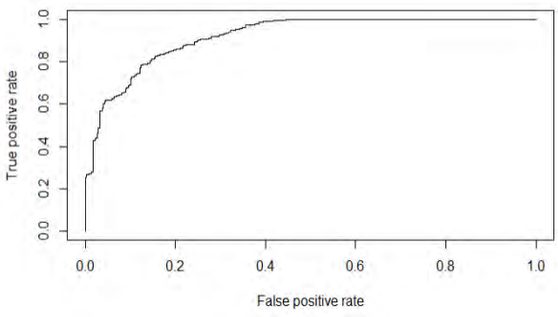


Figure 20. Diagram of ROC curve for FY2023 (Bronze Level, AUC:0.92).

8. RESULTS AND DISCUSSION OF LEARNING PROFICIENCY ANALYSIS

The mean and median values were calculated at the proficiency level for each year and are shown in Table 3, 4 (The equating is not necessary since the test content is the same). Since median of the scoring results have been increasing year by year, it can be said that the overall learning ability of the students has been improving year by year. This may be related to the introduction of compulsory programming education for elementary school students from the 2020 academic year, and for junior high school students from the 2021 academic year. It may be due to the fact that they are now consciously putting more effort into learning programming.

Table 3. Mean and median of scoring results.

	FY2021	FY2022	FY2023
Median(Entry Level)	-0.251	0.275	0.488
Mean(Entry Level)	-0.602	0.281	0.486
Median(Bronze Level)		0.948	1.024
Mean(Bronze Level)		0.875	0.822

9. RESULT AND DISCUSSION OF THE EQUATING METHODS

This section shows us the results and discussion of the equating method for difficulty and discrimination. As noted by Kato et al. (2014), equating methods are widely used and effective in practice. Although equating is often applied to learning proficiency, this study applies the Mean & Mean method to compare each item of the Entry and Bronze levels separately for the 2022 and 2023 administrations. According to Ishikawa (2022), 4 equating methods for common items—the Mean & Sigma method, the Mean & Mean method, Haebara's method, and the Stock and Lord method—were compared, with the Mean & Mean method being recommended as the preferred approach. For the items whose scales have been aligned, we examine their characteristics by plotting discrimination and difficulty on a 2-dimensional plane. These are shown in Figures 21 and 22, with the results of Mean & Mean method displayed separately for the FY2022 and FY2023. The green points represent the Bronze level, while the red points represent the Entry level. Let the F-set and the T-set represent the sets of items from two different tests, respectively. Consider the case in which the parameter θ of the F-set is linearly transformed to the scale of the T-set, resulting in $\theta^* : \theta^* = A\theta + K$. Determining the equating coefficients A and K corresponds to the equating. Using these coefficients, the item parameters after equating can be expressed as follows

$$a_{jT} = \frac{a_{jF}}{A}, \quad b_{jT} = Ab_{jF} + K$$

where a_{jF} and b_{jF} represent the discrimination and difficulty parameters, respectively, of item j in the F-set, and a_{jT} and b_{jT} represent the corresponding parameters of item j in the T-set. The equating coefficients obtained from the Mean & Mean method and the Mean & Sigma method are shown below (Table 4).

Table 4. The results of equating coefficients (Mean & Mean Method / Mean and Sigma Method).

	FY2022	FY2023
$\hat{A}$	0.86/1.35	1.32/1.39
$\hat{K}$	-0.51/-0.37	0.27/0.32

For the equating procedure, the coefficients were estimated from the 11 Scratch items identified as common to the Bronze and Entry levels. For example,

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

cases in which the "source" and "target" were the same were regarded as the same problem, even if their internal values differed.

Sample1(Bronze item): The display of the wizard(initial placement, when the flag is pressed.), The wizard size is set to [70] %.

Sample2(Entry item): The display of the fish(initial placement, when the flag is pressed.), The fish size is set to [50] %.

The 2 items shown above, Sample1 and Sample2, were regarded as having the same content. Although the values of the equating coefficients for 2022 differ between the 2 methods, their overall tendencies appear to be similar. In the sections that follow, the coefficients derived from the Mean & Mean method are used as the basis for interpretation.

Figure 21. Plotting Discrimination and Difficulty on a 2-dimensional plane for FY2022

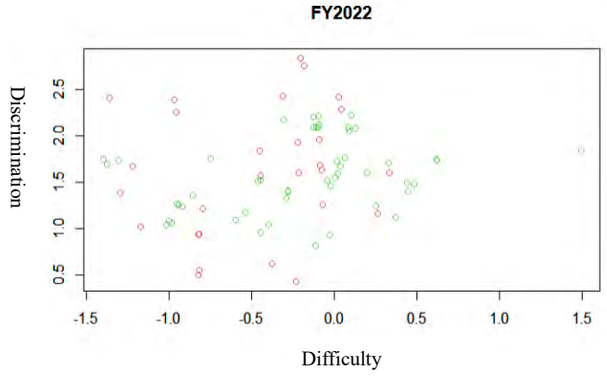
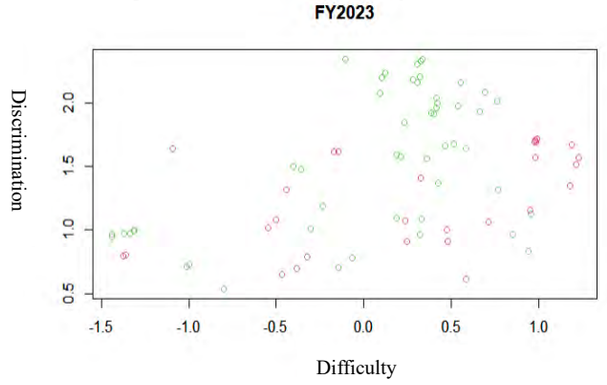


Figure 22. Plotting Discrimination and Difficulty on a 2-dimensional plane for FY2023



In Figure 21, several Bronze-level items with very high difficulty can be observed (The green points represent the Bronze level, while the red points represent the Entry level.). In contrast, in Figure 22, many of the Bronze-level items that show notable difficulty also exhibit high discrimination. As a result of this influence, items that demonstrated high difficulty in 2022 appear to have become somewhat less demanding in 2023, making them attainable for participants with higher levels of learning proficiency. In Table 3, for the Bronze level items, the median learning proficiency increased from FY2022 to FY2023, while the mean decreased. This suggests that overall learning ability has improved, but the disparity between higher and lower performers has widened. The presence of slightly more challenging items in FY2023 is also thought to be related to this effect.

10. CONCLUSION

In this study, we estimated parameters using Bayesian Item Response Theory (IRT) based on scoring results for different item contents at the Bronze and Entry levels, and examined their tendencies using equating methods. As a result, we demonstrated how equating methods can be applied to concrete cases in Scratch programming, as well as how to identify similar items. Furthermore, by performing equating, we were able to compare item suitability and difficulty across different problem items in the Entry and Bronze levels on a common scale. The observed tendencies included the presence of items with extremely high difficulty in the Bronze level in FY2022, and in FY2023, a larger number of Bronze-level items with high Discrimination compared to FY2022. These trends were presumed to be related to changes in the curriculum guidelines for programming education. In addition, the rise in the median learning proficiency in FY2023 compared with FY2022 suggests that overall learning has progressed. In addition, the increased participation of both higher- and lower-performing learners appear to have widened the gap between those who are proficient and those who are less proficient in programming. This is consistent with the observed decline in the mean learning proficiency for the Bronze level in FY2023. However, it should be noted that the present results are based on a sample from a limited geographic area. With programming education becoming mandatory, we would like to continue our research on what kind of changes will occur in the future and what factors will be responsible for such changes.

ACKNOWLEDGEMENTS

The authors declare that they have no conflicts of interest relevant to this study. This work was partly supported by JSPS KAKENHI Grant Number JP24K15230.

REFERENCES

[1]Ryota Kozakai, Tendency Analysis of Scratch Course (Bronze Level) for Junior High School Students, JSSE Research Report, Vol.39, No.3, pp.85-90, 2025-2.

[2]Ryota Kozakai, Wenxuan Zhang, Toshiki Kobayashi, Yuji Watanabe, Tendency Analysis in Scratch Programming Visiting Lecture at a Junior High School, IEICE Technical Report ET2023-1, pp.1-7, 2023.

[3]Ryota Kozakai, Qun Yang, Wenxuan Zhang, Toshiki Kobayashi, Yuji Watanabe, About Automatic Scoring of Google Colaboratory Python Programs, IEICE Technical Report ET2022-6, pp.20-25, 2022.

[4]Ryota Kozakai, Shoichiro Hara, Yuji Watanabe, Learning Tendency Analysis of Scratch Programming Course(Entry Class) for Upper Elementary School Students Based on Bayesian Item Response Theory, Proceedings of the Second International Conference of AI new Technology and open Discussion (ICAITD 2025), 2025-6.

[5]Fujitsu FOM Limited, Fun with Scratch 3.0 Let's! Programming Junior Programming Certification Official Textbook, FOM publication, 2019.

[6]Kanemune Susumu, Programming education to be made compulsory in elementary, junior high and high schools and the introduction to research, IEICE Communications Society Magazine, Vol. 13, No. 2, pp. 92-99, 2019.

[7]Sato Yukari, JSON Analysis for Learning Support of Scratch Programming, Shibata Gakuen Research Bulletin Vol.3 No.1, pp.13-24, 2023.

[8]Hideki Mori, Manabu Sugisawa, Kai Zhang, and Takanori Maesako, Practice of Elementary School Programming Class Using Scratch -Rethinking of Programming Education for Elementary School Students-, Transactions of the Japan Society for Educational Technology, 34(4), pp.387-394, 2011.

[9] Hideki Mori, Programming education for liberal arts college students using Scratch, Transactions of the Japan Society for Educational Technology, 34(Suppl.), pp.141-144, 2010.

[10] Makoto Nakazawa, Michitaka Aramoto, Masayuki Goto, Shigekazu Hirasawa, Learning Analytics via Visualization System of Edit Record −An Analysis of Learner's Thought Patterns for Elementary Programming Education using Scratch-, Proceedings of the 78th National Convention of Information Processing Society of Japan, pp.531-532, 2016.

[11] Hitoshi Susono, Eri Ohno, Maki Hagino, Kazunoh Enomoto, Scratch Programming for STEAM Learning in Elementary and Middle Schools, Mie University, Bulletin of the Faculty of Education, Volume 74 Number 1, Educational Practice, pp.151-158, 2022.

[12] [For Scratch beginners] Let's make a game “Run away from sharks”! Explaining how to make it, https://harusatoweb.com/blog/scratch/game9/#vk-htags-01fc62bc-f02a-401b-b42b-68dc6efb2a58

[13]Scratch - Imagine Program Share, https://scratch.mit.edu/.

[14]Kentaro Kato, Kazumitsu Kawabata, Item Response Theory in R, Ohmsha, 2014.

[15]Yuki Tsukihara, Keiichi Suzuki, Hideo Hirose, A small implementation case of the mathematics tests with the Item Response Theory evaluation into an e-learning system, Computers & Education VOL24, pp.70-76, 2008.

[16] Yasuhiro Ozaki, Tomoyuki Matsuzaka, Time Trace Analysis of Basic Mathematical Ability Using Item Response Theory, Bulletin of Hachinohe Institute of Technology, Vol.27, pp.61-67, 2007.

[17] Nobuhiro Oishi, Year-to-year Comparison of Small-scale Tests by Means of Bayesian IRT and Horizontal Equating, Kumamoto National College of Technology Research Bulletin No.14, pp.24-27, 2022.

[18] Herbert Robbins & Sutton Monro, A Stochastic Approximation Method, Annals of Mathematical Statistics, 22(3), pp.400–407, 1951.

[19] Gelman, A., Carlin, J.B., Stern, H.S., Dunson, D.B., Vehtari, A., & Rubin, D.B. (2013). Bayesian Data Analysis (3rd ed.). CRC Press.

[20] Vehtari, A., Gelman, A., Simpson, D., Carpenter, B., & Bürkner, P.C. (2021). Rank-normalization, folding, and localization: An improved R̂ for assessing convergence of MCMC. Bayesian Analysis, 16(2), 667–718.

received on Dec. 5th 2025
accepted on June 14th 2026

Appendix

The posterior distribution is given by

$$\prod_{i,j}\left(P(a_j,b_j,\theta_i)\right)^{\delta_{ij}}\left(1-P(a_j,b_j,\theta_i)\right)^{1-\delta_{ij}} \times \pi(a_j)\pi(b_j)\pi(\theta_i)$$

where

$$\pi(a_j)\sim N(1,1),\ \ \pi(b_j)\sim N(0,1),\ \ \pi(\theta_i)\sim N(0,1).$$

The Metropolis-Hastings method is used for posterior sampling. The acceptance probability is

$$\min\left(1,\ \frac{\frac{f(X_n|\vartheta_k^{j+1})\pi(\vartheta_k^{j+1})}{p(\vartheta_k^{j+1},\vartheta_k^{j})}}{\frac{f(X_n|\vartheta_k^{j})\pi(\vartheta_k^{j})}{p(\vartheta_k^{j},\vartheta_k^{j+1})}}\right)$$

The proposal probabilities are as follows

$$p(a_j^{K+1},a_j^K) \text{ is } a_j^{K+1}\sim N(a_j^K,1),$$
$$p(b_j^{K+1},b_j^K) \text{ is } b_j^{K+1}\sim N(b_j^K,1),$$
$$p(\theta_i^{K+1},\theta_i^K) \text{ is } \theta_i^{K+1}\sim N(\theta_i^K,1).$$

The initial values of a_j^0, b_j^0, θ_i^0 are uniformly set to 2, 2, and 1, respectively. A calculation step is

$$\vartheta_k^j \to (1-\gamma)\vartheta_k^j+\gamma\vartheta_k^{j+1},\ \ 0<\gamma<1.$$

A program for testing unidimensionality is also provided. When the standard Metropolis-Hastings (MH) algorithm ($\gamma=1$) is applied to imbalanced response data in which correct answers predominate, posterior sampling tends to converge toward a solution that maximizes the total log-likelihood—that is, a compromise solution biased toward the majority class (correct responses). Under this compromise, the discrimination parameter a_j is prone to underestimation, yielding a shallower sigmoid curve and thereby increasing the probability that examinees near the ability boundary are classified as correct. This results in reduced specificity, defined here as the capacity to correctly identify incorrect responses as such. MAP estimation, by contrast, adopts the mode of the posterior distribution as its point estimate, and consequently selects a steeper sigmoid—equivalently, a larger discrimination parameter a_j —necessary for reliably distinguishing the minority class (incorrect responses). This property follows mathematically from the skewness structure of the posterior distribution under the two-parameter logistic (2PL) model with imbalanced data. Specifically, a response pattern dominated by correct answers induces a right-skewed marginal posterior for a_j, such that the mode (MAP estimate) exceeds the posterior mean. It is therefore theoretically expected that MAP estimation yields a higher discrimination parameter and, in turn, improved specificity relative to estimation based on the posterior mean obtained via MH sampling. The sequential update rule with step size $\gamma=0.01$ adopted in the present study admits an interpretation within the stochastic approximation framework of Robbins and Monro (1951). The update scheme accumulates expected displacement in the direction of increasing posterior probability—that is, toward higher acceptance probability—and may thus be regarded as a candidate satisfying the Robbins–Monro convergence conditions, providing a theoretical basis for stochastic convergence to the neighborhood of the MAP solution after 1,000 to 20,000 or more iterations. Compared with the classical Robbins–Monro (1951) framework, the present method differs in three key respects. First, the Robbins–Monro method requires the exact unbiasedness condition, which guarantees that the expected update direction points precisely toward the root. In the present method, the update direction is governed by the MH acceptance probability, and the expected drift toward the MAP solution holds only approximately, under the assumption that the posterior distribution is unimodal. Second, the Robbins–Monro framework requires a monotonicity condition on the regression function, which plays the role of strong convexity by ensuring that the drift consistently points toward the root. In the present method, the analogous condition depends on the shape of the posterior; for the 2PL model with imbalanced data, the posterior is expected to be unimodal with its mode at the MAP estimate, but this is not guaranteed in general. Third, the Robbins–Monro convergence theorem requires that the step sizes a_n satisfy $\sum_{n=1}^{\infty}a_n<\infty$, a condition typically met by a decreasing schedule such as $a_n=1/n$. The constant step size γ = 0.01 adopted here violates this condition; consequently, the iterates do not converge to the exact MAP solution but to a neighborhood thereof, whose radius is proportional to γ. This is consistent with the empirically observed stabilization of log-posterior values after 20,000 iterations. For these reasons, the proposed method is more precisely characterized as a method inspired by, rather than strictly satisfying, the Robbins–Monro convergence conditions. In light of these properties, it is appropriate to characterize the proposed method as ***"Kozakai's gradient-free stochastic smoothing optimization"***, which inherits the proposal mechanism of MH while requiring no gradient information. By smoothing stochastic fluctuations in the

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

search process during convergence toward the MAP solution, the method exhibits intermediate characteristics between conventional MCMC sampling and MAP optimization, and is considered to offer an effective estimation strategy for discriminative learning under imbalanced data conditions.

Bayesian IRT Estimation - R Code

0. Data Loading

```{r data-loading}
library(openxlsx)
library(dplyr)

dat <- read.xlsx(
  "~/simulationData.xlsx",
  sheet = 1
)
dummy_dat_samples <- dat[, -1]
```

1. Unidimensionality Check

```{r unidim}  eval=FALSE
library(psych)

mat            <- dummy_dat_samples
polychoric.cor <- polychoric(mat)

print(polychoric.cor$rho)
VSS.scree(polychoric.cor$rho, main = "Scree Plot")
```

2. MCMC Sampler (with ESS / $\hat{R}$ Support)

```{r mcmc-setup}
set.seed(12345)

mat <- dummy_dat_samples
n   <- ncol(mat)   # number of items
N   <- nrow(mat)   # number of subjects

# ---- Log-likelihood ----------------------------
loglik <- function(p, num) {
  alpha <- p[1:n];  p <- p[-(1:n)]
  beta  <- p[1:n];  p <- p[-(1:n)]
  theta <- p

  mat_sub   <- mat[num, , drop = FALSE]
  # compute probability matrix (length(num) x n) in one shot
  logit_mat <- 1.7 * sweep(outer(theta[num], beta, "-"), 2, alpha, "*")
  pr        <- pmin(pmax(1 / (1 + exp(-logit_mat)), 1e-5), 1 - 1e-5)

  sum(mat_sub * log(pr)) + sum((1 - mat_sub) * log(1 - pr))
}

# ---- Log-posterior -----------------------------
sigma     <- 1
posterior <- function(x) {
  likelihood <- loglik(x, 1:N)
  aa <- x[1:n];  x <- x[-(1:n)]
  bb <- x[1:n];  x <- x[-(1:n)]
  ss <- x

  likelihood +
    sum(dnorm(aa, 1, sigma, log = TRUE)) +
    sum(dnorm(bb, 0, sigma, log = TRUE)) +
    sum(dnorm(ss, 0, sigma, log = TRUE))
}

# ---- MCMC settings -----------------------------
ite      <- 10000   # total iterations
burnin   <- 1000    # burn-in iterations
n_chains <- 3       # number of chains (required for R-hat)
eta      <- 10^(-2) # step size for partial update

# ---- Single-chain MH sampler (component-wise update) -------------------
run_chain <- function(seed, init_A, init_B, init_S) {
  set.seed(seed)
  A <- init_A;  B <- init_B;  S <- init_S

  A_samples  <- matrix(NA, nrow = ite, ncol = n)
  B_samples  <- matrix(NA, nrow = ite, ncol = n)
  S_samples  <- matrix(NA, nrow = ite, ncol = N)
  post_trace <- numeric(ite)

  for (l in 1:ite) {

    for (j in 1:length(A)) {
      # --- update discrimination alpha ---
      a0 <- rnorm(1, A[j], sigma)
      A0 <- A;  A0[j] <- a0

      r <- min(c(1,
        exp((posterior(c(A0, B, S)) - log(dnorm(a0,   A[j], sigma))) -
            (posterior(c(A,  B, S)) -
```

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

```
log(dnorm(A[j], a0,   sigma))))
      ))

      val <- sample(c(0:1), 1, prob = c(r, 1 -
r))
      if (val == 0) { A <- A * (1 - eta) + A0 *
eta }
      A[A < 0] <- 1e-5

      # --- update difficulty beta ---
      b0 <- rnorm(1, B[j], sigma)
      B0 <- B;  B0[j] <- b0

      r <- min(c(1,
        exp((posterior(c(A, B0, S)) -
log(dnorm(b0,   B[j], sigma))) -
            (posterior(c(A, B,  S)) -
log(dnorm(B[j], b0,   sigma))))
      ))

      val <- sample(c(0:1), 1, prob = c(r, 1 -
r))
      if (val == 0) { B <- B * (1 - eta) + B0 *
eta }
    }

    # --- update ability theta ---
    for (j in 1:length(S)) {
      s0 <- rnorm(1, S[j], sigma)
      S0 <- S;  S0[j] <- s0

      r <- min(c(1,
        exp((posterior(c(A, B, S0)) -
log(dnorm(s0,  S[j], sigma))) -
            (posterior(c(A, B, S )) -
log(dnorm(S[j], s0,   sigma))))
      ))

      val <- sample(c(0:1), 1, prob = c(r, 1 -
r))
      if (val == 0) { S <- S * (1 - eta) + S0 *
eta }
    }

    A_samples[l, ] <- A
    B_samples[l, ] <- B
    S_samples[l, ] <- S
    post_trace[l]  <- posterior(c(A, B, S))

    if (l %% 100 == 0)
      cat(sprintf("seed=%d  iter=%d
logpost=%.2f\n", seed, l, post_trace[l]))
  }

  list(A = A_samples, B = B_samples, S =
S_samples, post = post_trace)
}

# ---- Run multiple chains (different initial
values per chain) ----------
chains <- vector("list", n_chains)
for (k in 1:n_chains) {
  init_A      <- 2 + abs(rnorm(n, 1, 0.5))
  init_B      <- 1 + rnorm(n, 0, 1)
  init_S      <- 1 + rnorm(N, 0, 1)
  chains[[k]] <- run_chain(seed   = 1000 + k,
                           init_A = init_A,
                           init_B = init_B,
                           init_S = init_S)
}
```

3. ESS and R̂ Diagnostics

```
{r diagnostics}
library(coda)

# ---- Convert post-burnin samples to mcmc.list
--------------------------
to_mcmc <- function(chain) {
  post_mat <- cbind(chain$A, chain$B, chain$S)
  colnames(post_mat) <- c(
    paste0("alpha[", 1:n, "]"),
    paste0("beta[",  1:n, "]"),
    paste0("theta[", 1:N, "]")
  )
  mcmc(post_mat[(burnin + 1):ite, ], start =
burnin + 1, end = ite)
}
mcmc_list <- mcmc.list(lapply(chains, to_mcmc))

# ---- Effective Sample Size (ESS) ------------
--------------------------
ess <- effectiveSize(mcmc_list)
cat("=== ESS Summary ===\n")
print(summary(ess))
head(sort(ess), 10)   # top-10 parameters with
lowest ESS (high autocorrelation)

# ---- Gelman-Rubin R-hat ---------------------
--------------------------
rhat <- gelman.diag(mcmc_list, multivariate =
FALSE, autoburnin = FALSE)
cat("\n=== R-hat Summary (Point est.) ===\n")
print(summary(rhat$psrf[, "Point est."]))

bad <- which(rhat$psrf[, "Point est."] > 1.1)
cat("\nNumber of parameters with R-hat > 1.1:",
length(bad), "/", nrow(rhat$psrf), "\n")
if (length(bad) > 0) {
  print(rhat$psrf[bad, , drop =
FALSE][1:min(10, length(bad)), ])
```

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

```
}

# ---- Posterior trace plot per chain ---------
--------------------------
min_val <- min(c(chains[[1]]$post,
chains[[2]]$post, chains[[3]]$post))
max_val <- max(c(chains[[1]]$post,
chains[[2]]$post, chains[[3]]$post))

for (k in 1:n_chains) {
  plot(
    1:ite, chains[[k]]$post,
    xlim = c(1, ite), ylim = c(min_val,
max_val),
    col  = k,
    xlab = "Iterations",
    ylab = "log_posterior_values"
  )
  if (k != n_chains) par(new = TRUE)
}
```

4. Accuracy Evaluation Using Posterior Means

```
{r evaluation}
# use post-burnin samples combined across all
chains as point estimates
combine_mean <- function(field) {
  mats <- lapply(chains, function(ch)
ch[[field]][(burnin + 1):ite, , drop = FALSE])
  colMeans(do.call(rbind, mats))
}

A <- combine_mean("A")
B <- combine_mean("B")
S <- combine_mean("S")

mat1 <- mat;  mat2 <- 1 - mat;  matp <- mat
for (i in 1:N) {
  pr <- 1 / (1 + exp(-1.7 * A * (S[i] - B)))
  mat1[i, ] <- mat1[i, ] * ifelse(pr > 0.5, 1,
0)
  mat2[i, ] <- mat2[i, ] * ifelse(pr > 0.5, 0,
1)
  matp[i, ] <- pr
}

# ---- ROC / AUC ------------------------------
--------------------------

library(ROCR)
pred <- prediction(c(matp), c(mat))
perf <- performance(pred, "tpr", "fpr")
plot(perf)
auc  <- as.numeric(performance(pred,
"auc")@y.values)
cat("AUC =", auc, "\n")

# ---- Confusion matrix / Accuracy ------------
--------------------------
confusion_mat        <- array(0, dim = c(2, 2))
confusion_mat[1, 1]  <- sum(mat1)
confusion_mat[1, 2]  <- sum(mat)  - sum(mat1)
confusion_mat[2, 1]  <- sum(1 - mat) -
sum(mat2)
confusion_mat[2, 2]  <- sum(mat2)
print(confusion_mat)
cat("Accuracy =", (sum(mat1) + sum(mat2)) /
prod(dim(mat)), "\n")

# ---- Export results -------------------------
--------------------------
write.csv(
  cbind(chains[[1]]$S[ite, ],
chains[[2]]$S[ite, ], chains[[3]]$S[ite, ], S),
  "~/S.csv", fileEncoding = "CP932"
)
write.csv(
  cbind(chains[[1]]$A[ite, ],
chains[[2]]$A[ite, ], chains[[3]]$A[ite, ], A),
  "~/A.csv", fileEncoding = "CP932"
)
write.csv(
  cbind(chains[[1]]$B[ite, ],
chains[[2]]$B[ite, ], chains[[3]]$B[ite, ], B),
  "~/B.csv", fileEncoding = "CP932"
)
write.csv(ess, "~/ess.csv", fileEncoding =
"CP932")
```

Simulation Results

An example of the simulation results under the conditions described above is provided below.

0. Log-Posterior Values

Figure 23. Convergence of Log-Posterior Values During Iterative Computation ($\gamma = 0.01$)

```
>chains[[1]]$post[ite]
[1] -209.2794
>chains[[2]]$post[ite]
[1] -206.0541
>chains[[3]]$post[ite]
[1] -206.0915
```

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

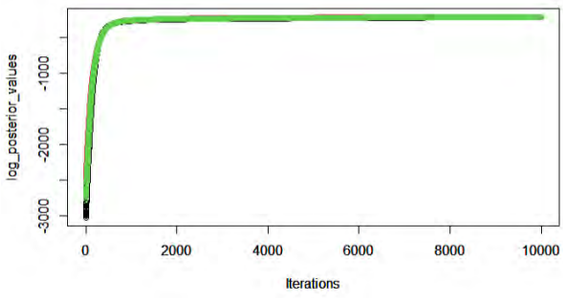
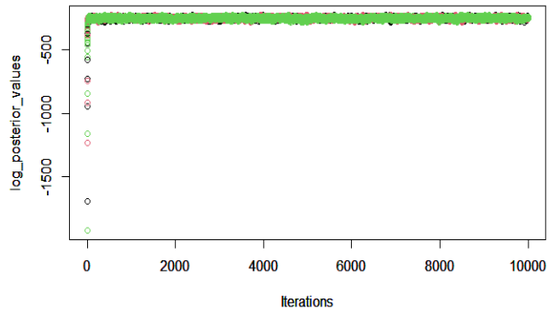


Figure 24. Convergence of Log-Posterior Values During Iterative Computation ($\gamma = 1$)

```
>chains[[1]]$post[ite]
[1] -239.0015
>chains[[2]]$post[ite]
[1] -242.986
>chains[[3]]$post[ite]
[1] -248.4228
> posterior(c(A,B,S))
[1] -205.3397
```



1. ROC Curves

Figure 25. Simulation Result of ROC Curve : Chain 1 ($\gamma = 0.01, AUC = 0.989$)

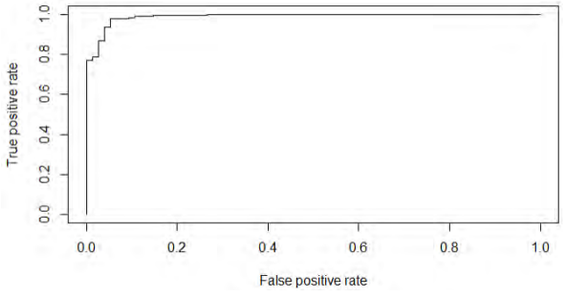


Figure 26. Simulation Result of ROC Curve : Chain 2 ($\gamma = 0.01, AUC = 0.989$)

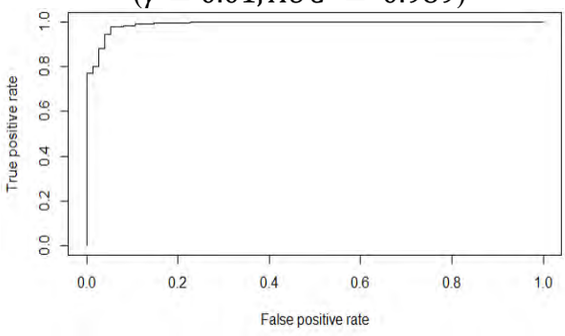


Figure 27. Simulation Result of ROC Curve : Chain 3 ($\gamma = 0.01, AUC = 0.989$)

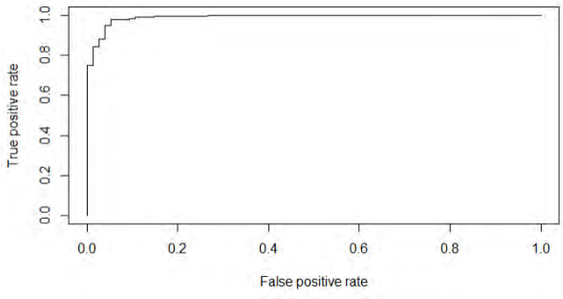
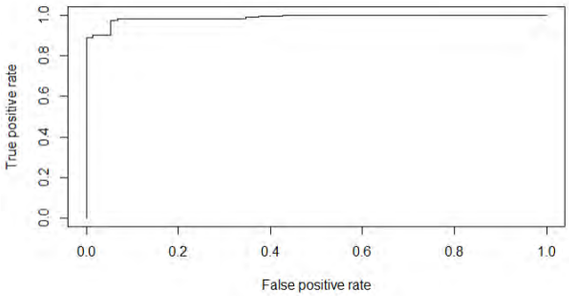


Figure 28. Simulation Result of ROC Curve ($\gamma = 1, AUC = 0.989$)



received on Dec. 5th 2025
accepted on June 14th 2026

2. Confusion Matrix

Table 5. The Result of Confusion Matrix ($\gamma = 0.01$, Chain 1)
- Accuracy: 0.9793
- Precision: 0.9862
- Recall: 0.9912
- Specificity: 0.8533
- F1 Score: 0.9887

$\gamma = 0.01$, Chain 1	Correct answer (Predict)	Incorrect answer (Predict)
Correct answer (observed)	787	7
Incorrect answer (observed)	11	64

Table 6. The Result of Confusion Matrix ($\gamma = 0.01$, Chain 2)
- Accuracy: 0.9804
- Precision: 0.9862
- Recall: 0.9924
- Specificity: 0.8533
- F1 Score: 0.9893

$\gamma = 0.01$, Chain 2	Correct answer (Predict)	Incorrect answer (Predict)
Correct answer (observed)	788	6
Incorrect answer (observed)	11	64

Table 7. The Result of Confusion Matrix ($\gamma = 0.01$, Chain 3)
- Accuracy: 0.9793
- Precision: 0.9862
- Recall: 0.9912
- Specificity: 0.8533
- F1 Score: 0.9887

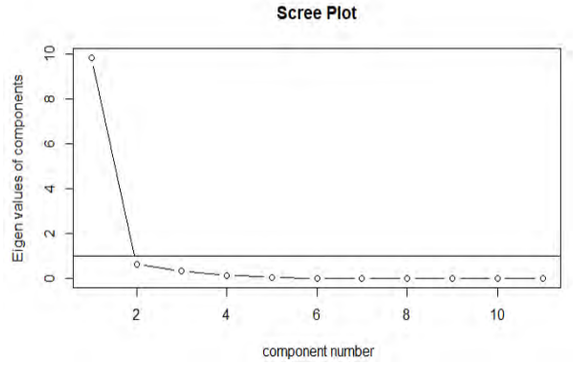
$\gamma = 0.01$, Chain 3	Correct answer (Predict)	Incorrect answer (Predict)
Correct answer (observed)	787	7
Incorrect answer (observed)	11	64

Table 8. The Result of Confusion Matrix ($\gamma = 1$)
- Accuracy: 0.9597
- Precision: 0.9623
- Recall: 0.9950
- Specificity: 0.5867
- F1 Score: 0.9783

$\gamma = 1$	Correct answer (Predict)	Incorrect answer (Predict)
Correct answer (observed)	790	4
Incorrect answer (observed)	31	44

3. The Scree Plot of the Eigenvalues

Figure 29. Simulation Result of Scree plot (Unidimensionality is confirmed.)



4. The Results of ESS (Effective Sample Sizes) and $\hat{R}$ ($\gamma = 1$, Burn-in:1000, 10000 Iteration.)

The convergence diagnostics used in this study, including the Gelman-Rubin statistic ($\hat{R}$) and effective sample size (ESS), are described in Gelman et al. (2013). Although all $\hat{R}$ values were below 1.01, indicating satisfactory inter-chain convergence, the minimum ESS was 906 (a_7, alpha[7].), corresponding to an efficiency of approximately 3.4% relative to the total post-burn-in draws (27,000). This suggests moderate autocorrelation for certain item discrimination parameters; nonetheless, the ESS of 906 remains above the minimum threshold of 400 recommended by Vehtari et al. (2021), indicating that the posterior estimates are reliable for inference.

```
=== ESS ===
> print(summary(ess))
   Min.     1st Qu.    Median
  906.2     3474.8    4230.5
```

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

```
   Mean     3rd Qu.      Max.
 3718.7     4439.8     4804.3

> head(sort(ess), 6)
alpha[7]   beta[7]   beta[2]
 906.186  1091.349  1178.415
beta[10]  theta[21]  beta[9]
 1363.81  1389.295  1410.471

=== R-hat (Point est.) ===
> print(summary(rhat$psrf[, "Point est."]))
   Min.     1st Qu.    Median
      1           1     1.001
   Mean     3rd Qu.      Max.
  1.001       1.001     1.004
```

Then it is reasonable to call it a **"point estimate"** — the code is implemented to return a single value for r, so the term is technically justified in this context.

5. Posterior Estimates of Item Discrimination $\hat{a}_j$ and Difficulty $\hat{b}_j$ Parameters

As shown for $\hat{a}_7$, the discrimination parameter estimated by the standard MH method ($\gamma = 1$) is relatively low, whereas the estimate under $\gamma = 0.01$ is the highest among all items (highlighted in red).

Table 9. The Results of $\hat{a}_j$, $\hat{b}_j$ $(\gamma = 1)$

Item	$\hat{a}_j$	$\hat{b}_j$
1	1.947	-1.744
2	2.345	-1.123
3	2.358	-1.682
4	2.056	-1.541
5	2.058	-2.197
6	1.732	-1.617
7	0.842	-0.609
8	2.348	-1.508
9	2.336	-1.356
10	2.353	-1.350
11	1.965	-1.419

Table 10. The Results of $\hat{a}_j$ $(\gamma = 0.01)$

Item	Chain 1	Chain 2	Chain 3
1	2.042	2.011	1.974
2	2.649	2.536	2.665
3	2.418	2.366	2.415
4	2.067	2.094	2.025
5	2.112	2.136	2.033
6	1.779	1.730	1.727
7	3.352	3.329	3.236
8	2.410	2.485	2.443
9	2.501	2.541	2.489
10	2.463	2.546	2.511
11	2.043	1.980	2.061

Table 11. The Results of $\hat{b}_j$ $(\gamma = 0.01)$

Item	Chain 1	Chain 2	Chain 3
1	-0.714	-0.755	-0.790
2	0.004	-0.073	-0.054
3	-0.640	-0.699	-0.694
4	-0.511	-0.536	-0.549
5	-1.198	-1.252	-1.268
6	-0.568	-0.631	-0.656
7	0.730	0.676	0.673
8	-0.434	-0.459	-0.481
9	-0.259	-0.318	-0.325
10	-0.272	-0.311	-0.311
11	-0.328	-0.426	-0.424

6. Posterior Estimates of Learning Proficiency θ_i

Table 12. The Results of $\hat{\theta}_i$ (MH : Metropolis-Hastings Method)

User	MH $\gamma = 1$	Chain1 $\gamma = 0.01$	Chain2 $\gamma = 0.01$	Chain3 $\gamma = 0.01$
1	0.017	0.516	0.498	0.473
2	0.011	0.489	0.448	0.451
3	-1.256	-0.233	-0.298	-0.293
4	0.581	1.334	1.236	1.173
5	0.011	0.560	0.467	0.452
6	0.026	0.481	0.464	0.423

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

7	0.599	1.237	1.203	1.296	45	0.012	0.508	0.460	0.466
8	0.585	1.294	1.226	1.216	46	0.599	1.207	1.205	1.178
9	-1.958	-0.939	-0.987	-1.007	47	0.018	0.538	0.426	0.464
10	0.616	1.316	1.221	1.178	48	0.601	1.323	1.193	1.286
11	0.006	0.493	0.441	0.457	49	0.617	1.285	1.174	1.157
12	0.573	1.219	1.212	1.188	50	0.603	1.212	1.240	1.172
13	0.595	1.303	1.189	1.227	51	0.611	1.224	1.206	1.239
14	0.597	1.214	1.235	1.222	52	0.590	1.241	1.185	1.194
15	0.609	1.276	1.238	1.264	53	0.011	0.504	0.434	0.463
16	0.000	0.486	0.442	0.444	54	0.579	1.217	1.209	1.209
17	0.602	1.241	1.230	1.198	55	0.603	1.225	1.224	1.187
18	0.010	0.500	0.444	0.452	56	0.012	0.486	0.446	0.424
19	0.003	0.486	0.468	0.473	57	0.595	1.191	1.268	1.210
20	0.625	1.261	1.247	1.169	58	-0.004	0.494	0.475	0.454
21	-1.349	-0.337	-0.401	-0.405	59	0.598	1.271	1.127	1.219
22	0.573	1.245	1.229	1.181	60	0.622	1.303	1.162	1.209
23	0.011	0.530	0.446	0.434	61	0.624	1.210	1.218	1.166
24	0.587	1.242	1.178	1.256	62	0.624	1.333	1.156	1.187
25	0.613	1.251	1.196	1.209	63	0.598	1.231	1.197	1.207
26	0.584	1.248	1.184	1.128	64	0.614	1.201	1.147	1.199
27	-1.985	-0.960	-1.017	-1.040	65	0.607	1.240	1.212	1.257
28	-2.626	-1.679	-1.771	-1.750	66	0.609	1.172	1.182	1.220
29	0.587	1.201	1.204	1.187	67	0.605	1.221	1.228	1.230
30	0.605	1.346	1.207	1.173	68	0.001	0.512	0.456	0.455
31	0.584	1.264	1.182	1.213	69	0.598	1.225	1.185	1.193
32	0.605	1.288	1.217	1.194	70	0.597	1.191	1.201	1.178
33	0.599	1.241	1.161	1.185	71	0.600	1.254	1.178	1.194
34	-2.626	-1.728	-1.743	-1.734	72	0.582	1.261	1.185	1.168
35	0.615	1.192	1.245	1.232	73	0.601	1.291	1.236	1.214
36	0.603	1.290	1.185	1.182	74	-0.429	0.693	0.622	0.652
37	0.013	0.480	0.474	0.410	75	-1.114	0.063	-0.033	0.008
38	0.613	1.272	1.169	1.315	76	-0.522	0.609	0.559	0.516
39	0.017	0.496	0.473	0.500	77	-0.784	0.473	0.412	0.389
40	0.622	1.295	1.291	1.236	78	0.590	1.309	1.170	1.173
41	0.602	1.261	1.169	1.251	79	0.582	1.191	1.181	1.134
42	0.612	1.225	1.246	1.224					
43	0.010	0.529	0.458	0.487					
44	0.607	1.228	1.215	1.173					

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

7. Sample Dataset

As shown in Table 13, the sample data exhibit a class imbalance, with correct answers substantially outnumbering incorrect answers. Table 14 presents the dataset used in this simulation.

Table 13. Number of correct and incorrect answers in the sample data

Item	The number of correct answers	The number of Incorrect answers
1	75	4
2	71	8
3	75	4
4	74	5
5	77	2
6	74	5
7	55	24
8	74	5
9	73	6
10	73	6
11	73	6

Table 14. Sample Dataset
($\boldsymbol{N} = \mathbf{79\ users}, \boldsymbol{n} = \mathbf{11}$ items)

1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
0	0	1	1	1	0	0	1	1	1	0
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	1	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	0	1	1	1	1	0	0	0	0	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1

https://doi.org/10.63211/j.p.25.260701

received on Dec. 5th 2025
accepted on June 14th 2026

1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	0
1	0	1	1	1	1	1	1	0	0	1
1	0	1	1	1	1	1	1	1	1	1
1	1	1	0	1	0	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1