Recommendations for Strategic Time Allocation and Performance Improvement Among Low-Ability Students in 15-213

Ein Jeong
2025

Table of Contents

I. Introduction

A. Background

This study examines how students behave and perform in Carnegie Mellon University's core undergraduate course, 15-213: Introduction to Computer Systems. Multiple-choice and short-answer questions on computer-based midterm and final exams were given to students from 2016 to 2019 to gauge their knowledge of operating systems, architecture, and systems programming.

Because these tests were given digitally, the exam platform was able to record detailed information on each student's activity, including when and how long they spent looking at each question, how frequently they went back to a particular question, and how their scores progressed over time. With the use of this data, researchers can reenact the test-taking procedure and investigate students' knowledge as well as their strategic exam-taking methods.

While earlier studies have assessed validity and fairness in randomized testing settings, this work focuses on how students manage their time, especially low-ability students who frequently struggle the most with scheduled tests. We hope to determine whether particular behavioral patterns – particularly how students allocate their time among the "Skim," "Solve," and "Review" phases – can have a significant effect on performance by utilizing clickstream data.

In order to improve educational evaluation and identify strategies that could help students who struggle in the course, this investigation adds to an expanding collection of research that employs digital trace data.

B. Classification of Ability Groups

In this study, students are divided into ability groups to investigate how performance level affects test-taking behavior. This model was based on Abigail Reese's research [1] on Item Response Theory (IRT) and statistical fairness in randomized computer-based examinations for 15-213. The Generalized Partial Credit Model (GPCM), which allocates a latent ability score to each student based on their performance across many question categories, is used in her work to estimate student ability. These estimates provide a psychometrically based assessment of ability by allowing predictions of how students would perform on different exam variations. In particular, the ntile() function is used to divide the population into:

- Low-ability students (bottom 1/3),
- Medium-ability students (middle 1/3), and
- High-ability students (top 1/3).

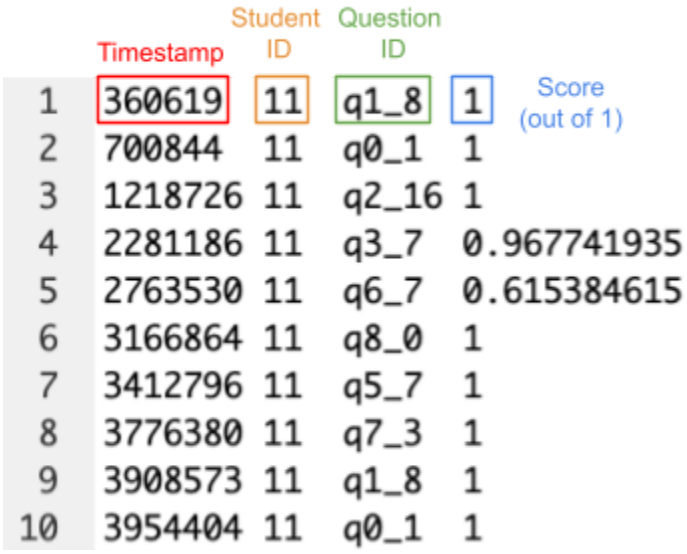
Prior to analysis, data cleaning criteria were used to make sure that these groupings accurately reflect significant behavior patterns. I excluded any student logs with negative timestamps, which were presumed to be corrupted log files. I also eliminated logs for students who did not view all of the given questions (7 for the midterm and 8 for the final), assuming that incomplete sessions do not represent real test-taking behavior. The final dataset only included students with complete question coverage and valid time data. These data filtering processes made sure that the ability groups were based on representative behavioral data. This makes comparisons between students of low, medium, and high ability more trustworthy in determining the relationship between time management strategies and performance.

II. Methodology

This study adopts a quantitative research approach to investigate test-taking behaviors among different student ability groups. The goal is to determine the relationship between exam performance and timing patterns (such as skimming, solving, and reviewing), especially for students who have limited ability. Since the dataset is time-based and numerical, statistical modeling and visualization approaches are suitable for identifying patterns and deriving conclusions that can be applied generically.

The primary research topic is: *How do different test-taking strategies affect the final score, especially for students in the low-ability group?* Specific sub-questions include whether early skimming provides a measurable benefit and how time allocation differs by ability group.

Figure 1: *Example Snippet from the Raw Exam Interaction Log*



	Timestamp	Student ID	Question ID	Score (out of 1)
1	360619	11	q1_8	1
2	700844	11	q0_1	1
3	1218726	11	q2_16	1
4	2281186	11	q3_7	0.967741935
5	2763530	11	q6_7	0.615384615
6	3166864	11	q8_0	1
7	3412796	11	q5_7	1
8	3776380	11	q7_3	1
9	3908573	11	q1_8	1
10	3954404	11	q0_1	1

The dataset [2] is derived from a computer-based exam platform that automatically collected detailed student interactions over assessments. As seen in Figure 1, each log entry contains the following information: the time since the start of the student's exam, the corresponding question identifier, the score assigned, and an anonymous student number. These records were analyzed to create per-student timelines that depict behavioral phases such as skimming, solving, and reviewing. This framework allowed for the study of cumulative performance trends throughout the exam.

I divided student activity into three behavioral phases and calculated the intervals between subsequent logs in order to extract this information. Additional features were created to compute progression-per-question, cumulative scores, normalized time proportions, and total and per-phase timings. Using quantile bins based on final results, students were then divided

according to ability, enabling comparisons across graded performance levels. There were no personal identifiers used, assuring complete compliance with ethical research guidelines.

This study mainly used statistical techniques in R and data visualization (such as ggplot2) to find patterns in the behavior of low, mid, and high-performing categories. To identify the best teaching methods for students with low ability, non-linear correlations between skimming time and overall score have been calculated using Generalized Additive Models (GAMs).

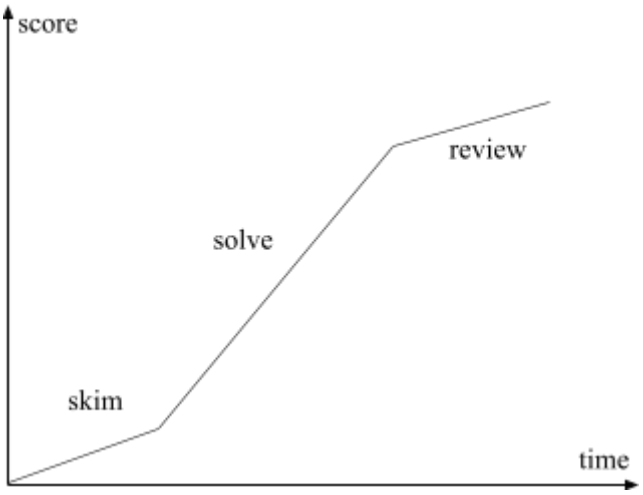
One advantage of this methodology is its ability to precisely model student behavior and its high data precision. Yet, the results may be affected by noise in the raw log data, as well as students' prior knowledge and test-taking strategies that were not recorded in the logs. Also, the researcher’s heuristic-based understanding of behavioral stages may not fully capture the complexity of students' cognitive processes. To mitigate these limitations, non-parametric models were employed when applicable.

III. Results

A. Identification of Exam Phases: Skim, Solve, and Review

To better understand how students spend their time during an exam, I divided their behavior into three phases: skim, solve, and review. These stages represent typical approaches that students could use to difficult problems: initially scanning the questions (Skim), actively working to find answers (Solve), and then going back to look for potential improvements (Review).

Figure 2: *Conceptual Representation of Skim, Solve, and Review*



Conceptually, skimming refers to the initial exchanges in which pupils are examining the test without making an urgent effort to solve the issues. The score-over-time curve has a low slope during this period because students often put in very little effort and achieve very little in the way of points. Solving occurs when students are actively engaged, as seen by a rapid increase in score over time. At this point, the student is working effectively and making the most progress in relation to the amount of time spent. Reviewing occurs later, when, despite ongoing time investment, score increases start to diminish, indicating declining returns and an outcome to the curve's low slope. This is seen in Figure 2.

To more precisely identify these phases, I examined the first derivative of the smoothed cumulative score progression curve, which was organized by ability level. The amount of score acquired per unit of time is indicated by the slope of this curve. The peak of this curve, where the slope is at its greatest, is known as the Solve phase. The point before this peak, where the slope first approaches 25% of the maximum (i.e., where score gain begins to accelerate), marks the end of the Skim phase. When the slope returns to 25% of the peak, indicating decreased scoring efficiency, the Review phase starts.

The use of a 25% threshold is based on conventional methods in time-series analysis and event-related potential (ERP), where researchers identify onset and offset regions in waveform data using fractional area metrics, frequently at 25% [3]. By removing noise and expanding the desired phase bounds, this approach assists in locating real change sites. Additionally, I tested 20% and 30% thresholds; full comparisons are provided in Appendix A.

B. Defining Meaningful Question Interactions

Understanding why higher-ability students outperform the lower-ability group begins by looking at how scores accumulate over time. However, in order to interpret time-based behavior meaningfully, we must first define what constitutes a substantive contact with the issue. The dataset consists of timestamped records of students clicking on exam questions, with each occurrence marked as a "view." However, not all perspectives reflect true cognitive effort—some may result from rapidly scanning questions at the beginning of the test or returning shortly out of interest without interacting with the content. To eliminate these non-informative views, I used two statistical methods to find suitable criteria for separating actual involvement from passive or inconsequential clicking.

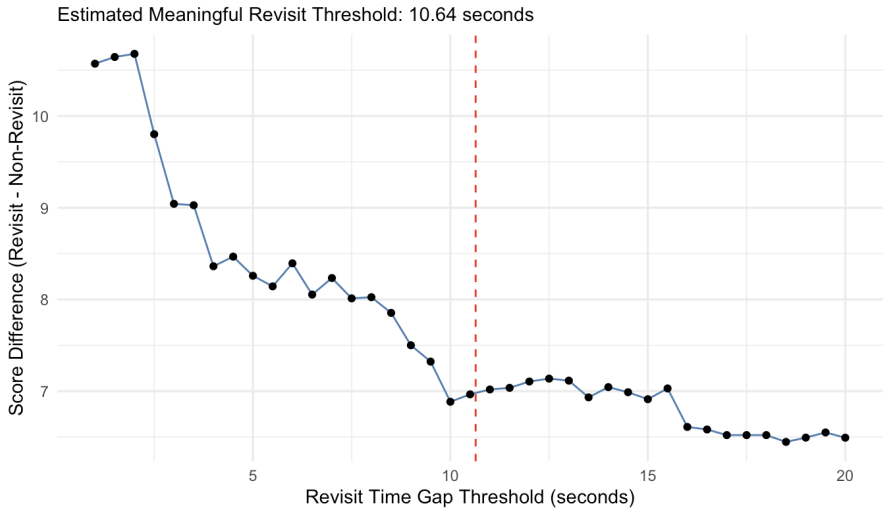
Figure 3: *Effect Size (Cohen's d), Mean Score Difference, and p-value Across Varying Minimum View Time Thresholds*

Threshold (seconds)	Mean Difference	p-value	Cohen's d
1	-0.05028854	3.007693e-81	-0.12730344
2	-0.01793733	5.337689e-07	-0.04498573
3	0.01878515	1.060716e-04	0.04740861
4	0.04178394	1.315342e-10	0.10644011
5	0.04618003	2.195748e-07	0.11735376
6	0.05618625	1.495913e-05	0.14263395
7	0.05174206	6.315994e-03	0.13185753
8	0.09526909	8.235664e-04	0.24577532
9	0.18201632	2.306459e-06	0.49355012
10	0.13059309	3.001155e-02	0.34930902

To determine a threshold for meaningful engagement, an investigation was performed over a range of minimal viewing durations. For each duration, performance differences were compared between students who saw questions longer than the threshold and those who did not. Figure 3 depicts the impact size (Cohen's d), mean score difference, and statistical significance (p-value) at each level. The impact size peaked at 11 seconds (Cohen's d = 0.624), with students who read

items for at least this long scoring 0.217 points higher ($p = 0.0026$). Conversely, durations of less than 3 seconds were related to minimal or negative impact sizes, indicating a lack of genuine cognitive involvement. Based on this tendency, a conservative threshold of 3 seconds was set to exclude views that were too brief to show intentional interaction.

Figure 4: *Score Difference Between Revisiting and Non-Revisiting Students Across Varying Revisit Time Gap Thresholds*



Simultaneously, I examined the distance between successive perspectives on the same subject to ascertain what makes a revisit significant. Figure 4 depicts the score difference between students who reviewed questions and those who did not, over a variety of time gaps. I chose the first point (after 2.5 seconds) where the absolute slope of the score difference curve falls below 0.01 in order to determine the "knee point" where score disparities halt. This translates to a revisit threshold of 10.64 seconds, beyond which the scores of students who revisit and those who do not stabilize. This suggests that while shorter revisit gaps are probably insignificant or impulsive, students who return to a question after a pause of at least 10.64 seconds are doing so with purpose.

As a result of these findings, I cleaned the dataset using the following criteria:

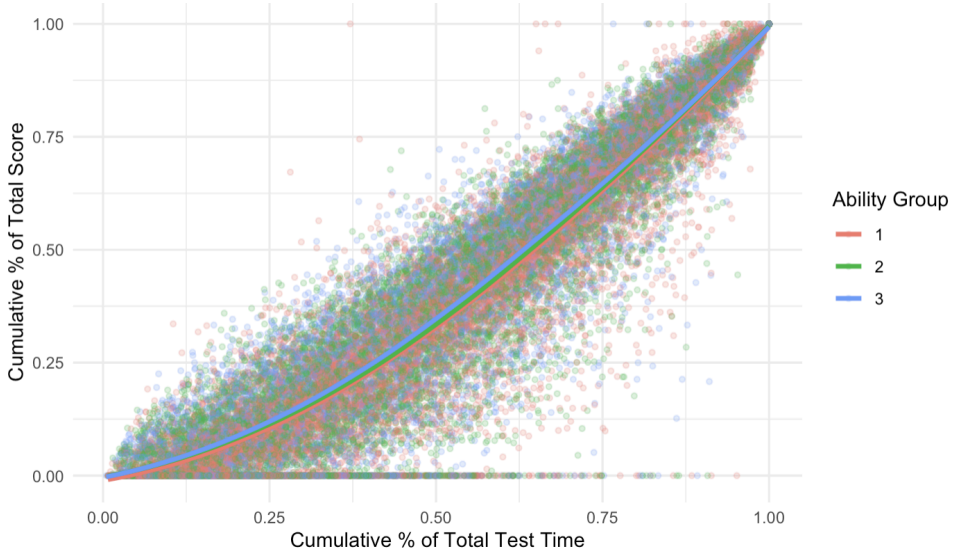
- I excluded any view events shorter than *3 seconds*, assuming they do not accurately represent cognitive processing.
- I also eliminated revisits with time gaps below *10.64 seconds*, as these are unlikely to represent significant review behavior.

These thresholds serve as the foundation for the behavioral phase analysis that follows and ensure that my analysis catches genuine student engagement rather than noise from surface-level navigation activity (Skim, Solve, Review).

C. Temporal Trends in Score Progression

After defining and clearing important question views at the 3-second and 10.64-second criteria, I examined how students' scores changed during the exam. I accomplished this by grouping students according to their ability level and graphing each student's cumulative percentage of total score against their cumulative percentage of total test duration. Figure 5 displays the resulting visualization.

Figure 5: *Cumulative Percentage of Total Score vs Cumulative Percentage of Total Test Time, Separated by Ability Group*



The LOESS-smoothed curves show the average score accumulation trajectory for each of the three ability groups, while each dot represents a single data point from a student-question interaction. Low-ability students (Group 1) are represented by red, medium-ability students by green, and high-ability students by blue (Group 3). All three groups' progression curves have a similar overall form, which is concave upward and increases gradually toward 100%. This suggests that most students, regardless of ability, earn most of their score close to the middle and end of the test.

Although the general slope trend of score progression seems to be concave upward for all groups, there are a few small differences. The blue curve consistently exceeds the red and green curves, particularly in the early and middle sections of the test, indicating that high-ability pupils typically receive their points earlier in the test. Low-ability students, on the other hand, gain points more slowly and have a curve that lags somewhat behind. This implies that while test-taking results are generally the same for all groups, high-achieving students might be more effective in their early involvement, answering questions faster or more accurately in the test's early phases.

D. Comparative Analysis Across Ability Groups

Figure 6: *Proportion of Total Test Time Spent in Skim, Solve, and Review Phases by Ability Group (1 = lowest ability, 3 = highest ability)*

Ability Group	Skim	Solve	Review
1	0.1716907	0.6004586	0.2688087
2	0.2094033	0.5323870	0.3008797
3	0.1928371	0.5413764	0.3263285

Using the previously described framework, Figure 6 shows the proportion of total test time each ability group spent in the Skim, Solve, and Review phases. The table reveals a number of significant patterns. First, review time rises with ability: students with higher ability levels take longer (around 4% of total time more) to go over questions again, perhaps to proofread or polish their responses. Second, solve time decreases with ability, from 60.0% in Group 1 to 54.1% in Group 3, indicating greater efficiency in working through questions. Lastly, skim time continues to rise with ability groups from 17.2% in Group 1 and 19.3% in Group 3, although it seems to be a little higher for the medium group (20.9%).

Figure 7: *ANOVA Test p-values for Differences in Phase Time Allocation Across Ability Groups*

Phase	p-value
Review	$2.514687e^{-10}$
Skim	$5.221971e^{-14}$
Solve	$3.593465e^{-23}$

I used an ANOVA test to compare the mean percentage of time spent in each phase across ability groups in order to evaluate whether these differences are significant statistically. All three phases – Skim, Solve, and Review – have very significant p-values (all less than 10^{-9}), demonstrating that the variations in time allocation across ability groups are not the result of chance. The results are displayed in Figure 7.

Figure 8: *Distribution of Time Spent in Skim, Solve, and Review Phases by Ability Group*

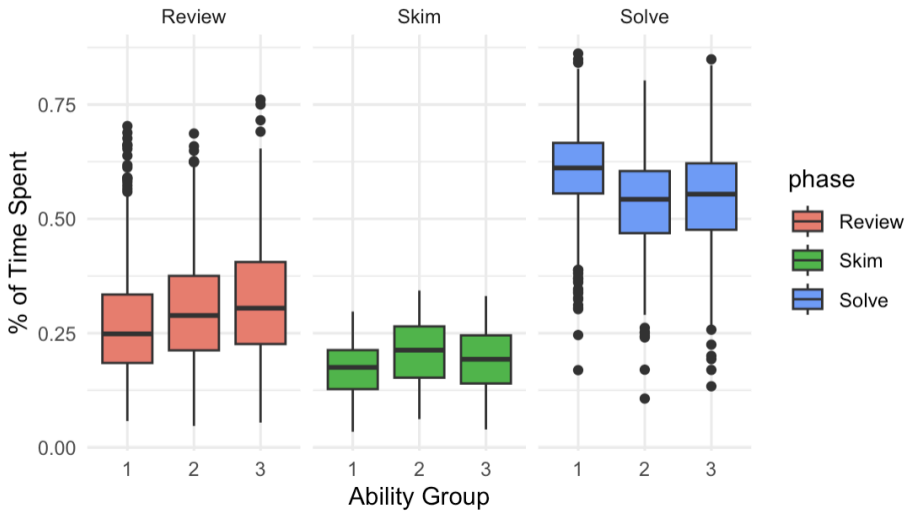


Figure 8 depicts this separation with a boxplot representing the distribution of time spent in each phase among ability groups. Higher-ability students exhibit a more balanced approach, spending more time reviewing and somewhat less time solving, while lower-ability students spend the most time solving and the least time skimming or reviewing, according to the median values.

Together, these results imply that students who perform well not only solve problems more accurately but also more quickly, spending more time going over their answers. This pattern's consistency offers compelling evidence that significant strategic distinctions can be identified by phase-based behavior analysis.

Figure 9: *Linear Regression Predicting Ability Score from Time Allocation in Each Phase*

```
Call:
lm(formula = ability_base ~ Review + Skim + Solve, data = time_allocation_wide)

Residuals:
     Min       1Q   Median       3Q      Max
-2.91034 -0.48844 -0.03877  0.54181  2.03437

Coefficients: (1 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -1.1590     0.1226  -9.455  < 2e-16 ***
Review        2.1761     0.2543   8.558  < 2e-16 ***
Skim          2.8162     0.4170   6.754 2.46e-11 ***
Solve             NA         NA      NA       NA
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.8036 on 990 degrees of freedom
  (316 observations deleted due to missingness)
Multiple R-squared:  0.08323,   Adjusted R-squared:  0.08138
F-statistic: 44.94 on 2 and 990 DF,  p-value: < 2.2e-16
```

I performed a multiple linear regression model using the proportion of time spent in each phase (Skim, Solve, and Review) to predict students' ability scores to confirm whether these behavioral variations translate into performance outcomes (Figure 9). Holding Skim constant, increasing time spent on Review (compared to Solve) considerably increases predicted ability, with a coefficient of 2.1761 ($p < 2e^{-16}$). Similarly, holding Review constant, spending more time Skimming (compared to Solve) is positively associated with ability, with a larger coefficient of 2.8162 ($p \approx 2e^{-11}$). These findings imply that while both activities imply higher-achieving students, skimming seems to be an even more reliable indicator of ability. As a result, I decided to concentrate my investigation on the skimming habits of low-ability students as they had the greatest impact on expected ability scores.

Figure 10: *Negative Correlation Between Solve Time and Ability Score*

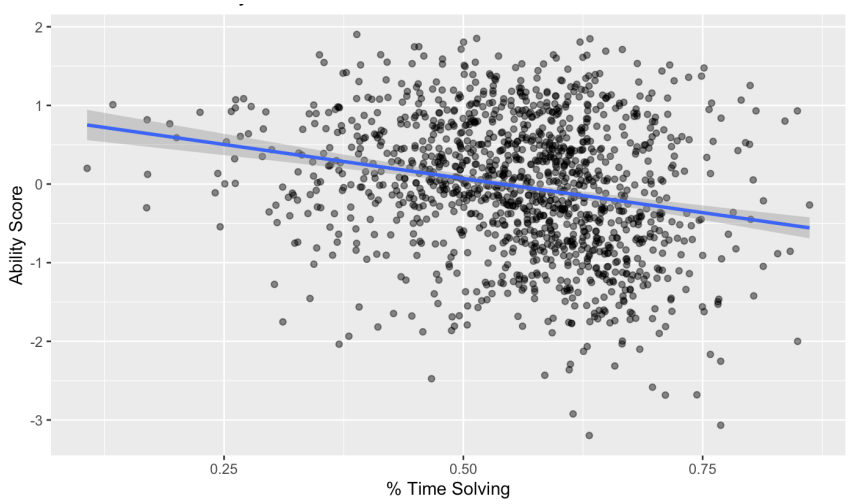


Figure 10's extrapolation plot confirms that students who devote a greater percentage of their time to solving problems typically fall into lower-ability groups. In the context of the 15–213 exams, this shows a definite negative correlation between solve time and ability level, even though it does not prove a causal relationship (i.e., we cannot conclude that spending more time solving causes lower performance).

E. The Predictive Role of Skimming Time in Low-Ability Students

To understand whether Skim time contributes to better performance among low-ability students, I conducted multiple statistical analyses.

Figure 11: *Positive Correlation Between Skim Time and Total Score Among Low-Ability Students (Pearson correlation: r = 0.3168, p < 1e-9)*

```
[1] Correlation: Skim Time vs Total Score (Low Ability)

        Pearson's product-moment correlation

data:  low_only$Skim and low_only$score_total
t = 6.2488, df = 350, p-value = 1.202e-09
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.2195457 0.4078318
sample estimates:
      cor
0.3168065
```

Figure 11 depicts the results of a Pearson correlation test between skim time and total score, which was confined to students with the lowest ability level. The test shows a statistically significant positive relationship ($r = 0.3168$, $p < 1e^{-9}$), suggesting that students with lower ability tend to score higher overall when they spend more time skimming.

Figure 12: *Low-Skim Students Score Lower Than High-Skim Students Even After Controlling for Total Time (ANCOVA: Δ = -0.52, p = 0.000626)*

```
[3] ANCOVA: Score ~ Skim Group + Total Time (Low Ability)

Call:
lm(formula = score_total ~ skim_group + total_time, data = low_only)

Residuals:
     Min       1Q   Median       3Q      Max
-2.99830 -0.60582  0.08604  0.67174  2.44242

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       4.4960237  0.1637061  27.464  < 2e-16 ***
skim_groupLowSkim -0.5223426  0.1513412  -3.451 0.000626 ***
total_time         0.0023363  0.0007727   3.023 0.002685 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.072 on 349 degrees of freedom
Multiple R-squared:  0.1485,    Adjusted R-squared:  0.1437
F-statistic: 30.44 on 2 and 349 DF,  p-value: 6.506e-13
```

However, correlation by itself does not demonstrate that performance is directly impacted by skim time. I used an ANCOVA model to determine whether skim time predicts scores while taking into consideration the overall amount of time students spend on the test (Figure 12). Students in this group were classified into HighSkim and LowSkim subgroups based on the median of their skim time distribution, with LowSkim defined as students in the bottom 50% of skim time within the low-ability group. The findings suggest that LowSkim students scored on average 0.52 points lower than HighSkim students ($p = 0.000626$), even after considering total time. This result confirms that the amount of time spent skimming has an independent, statistically significant effect on low-ability students' performance.

Figure 13: *Skim Time and Total Time Are Almost Perfectly Correlated (Pearson correlation: $r \approx 0.97$)*

```
Python
> cor(low_only$Skim, low_only$total_time)
0.9739871
```

Lastly, Figure 13 demonstrates an almost perfect correlation ($r = 0.97$) between skim time and overall test time. This implies that students who skim the test more often take longer overall. This serves as a warning because too much skimming might result in poor time management, which makes it more difficult to finish the test.

All of these results point to skim time as a crucial differentiator for students with low ability. Better performance seems to be linked to spending more time skimming, but only if this doesn't affect the overall tempo.

IV. Discussion

A. Optimizing Exam Strategies for Low-Ability Students

Figure 14-1: *GAM Estimating the Optimal Skim Time Ratio*

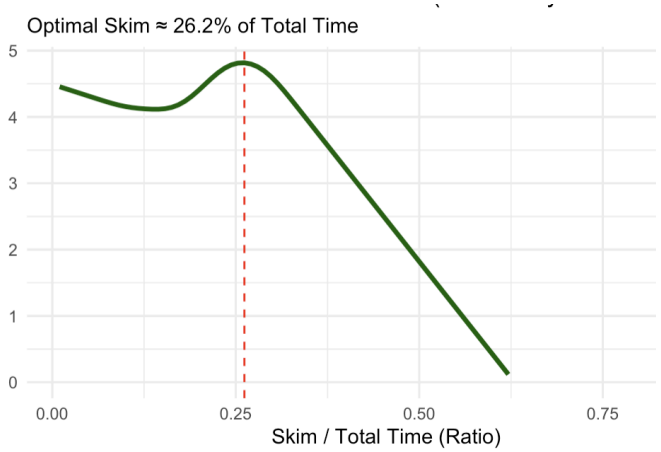
```
Parametric coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.46742    0.05878      76   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                edf Ref.df     F  p-value
s(Skim_ratio) 3.733  4.653 4.994 0.000346 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.0567   Deviance explained = 6.62%
GCV = 1.3122  Scale est. = 1.2956    n = 375

[Optimal Skim Ratio] 0.26 (i.e., 26.2% of total time)
```

Figure 14-2: *Predicted Total Score by Skim Time Ratio Among Low-Ability Students*



To establish practical instructions for low-ability students, I examined how the fraction of time spent skimming, denoted as the Skim Time Ratio, corresponds to overall exam performance. A smooth curve was fitted using a Generalized Additive Model (GAM), as illustrated in Figure 14-2, to capture the non-linear relationship between scores and skimming. The greatest projected score occurs at a Skim Time Ratio of roughly 0.26, or 26.0% of total test time, according to the model, which shows a statistically significant effect ($p < 0.001$) (Figure 14-1).

This implies that the best approach for students with low ability may be to devote approximately one quarter of exam time to skimming. However, as the graph illustrates, even if students skim for a little shorter time, performance is comparatively constant, suggesting a level of robustness around the 26% barrier. There may be a penalty to over-skimming, though, as performance tends to deteriorate significantly after this.

There are a few limitations that are worth mentioning. First, it is challenging to purposefully achieve a precise skimming ratio because students are unaware of their entire exam time beforehand. Second, since fewer students spend so much time skimming, the sharp drop in performance at 26% might be the result of noise or sparsity in the dataset. Last but not least, this approach does not differentiate between difficult and easy questions, which may affect the effectiveness of skimming if question difficulty consistently varies.

The suggestion is supported by an ANCOVA analysis (Figure 12), which demonstrates the value of strategic skimming at the beginning of the test. Even after adjusting for total time, low-ability

students who skim properly (spending around 26%) score on average 0.52 points higher (on a 0–100 scale) than their peers who skim poorly.

B. Application to Midterm Exam Dataset

To see if the Skim-Solve-Review structure applies beyond the final exam, I ran the same analysis on the midterm data. The cumulative score development over time, as illustrated in Figure 15-1, follows a similar trend to the final exam, with the majority of students, irrespective of ability group, gaining more points as time passes (concave up). This implies that both tests capture the general temporal patterns of problem-solving behavior.

Figure 15-1: *Cumulative Percentage of Total Score vs Cumulative Percentage of Total Test Time, Separated by Ability Group*

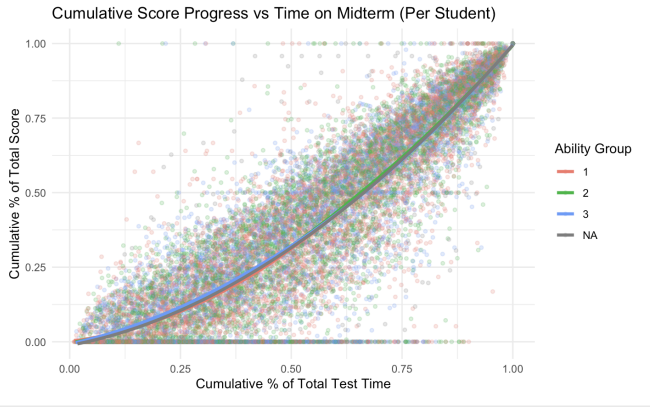


Figure 15-2: *Effect of Skim Group and Total Time on Midterm Score Among Low-Ability Students*

```
[3] ANCOVA: Score ~ Skim Group + Total Time (Low Ability, Midterm)

Call:
lm(formula = score_total ~ skim_group + total_time, data = low_only_mid_labeled)

Residuals:
    Min      1Q  Median      3Q     Max
-3.6487 -1.0755  0.0123  0.9690  3.5286

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     3.516754   0.233849  15.039   <2e-16 ***
skim_groupLowSkim -0.470450   0.216186  -2.176   0.0303 *
total_time      0.001970   0.001108   1.778   0.0765 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.428 on 305 degrees of freedom
  (44 observations deleted due to missingness)
Multiple R-squared:  0.06984,   Adjusted R-squared:  0.06374
F-statistic: 11.45 on 2 and 305 DF,  p-value: 1.604e-05
```

According to the ANCOVA results in Figure 15-2, low-skim students scored 0.47 points lower on the midterm than their high-skim peers, even after accounting for total test time. This effect is marginally less pronounced than the 0.52 point difference noted in the final test (see Section III.E), but it is still significant ($p = 0.0303$). This discrepancy implies that while skim time is still a predictor of performance, its impact may change based on the exam's structure or stakes.

Figure 15-2 also indicates that the overall model explains less variance (Adjusted $R^2 = 0.0637$), suggesting that other factors may be more important in the midterm. Yet, the consistency in direction and statistical significance across both tests confirms the strength of the skim time performance association, especially among lower-ability students.

V. Threats to Validity

Although this study offers compelling evidence that students with lower ability levels who spend more time skimming typically perform better, a number of factors could compromise the validity and generalizability of these findings. First, even with statistical controls in place, the results are still correlated, meaning that we cannot draw conclusions about a causal relationship between better performance and more skimming. It's possible that both behaviors and results are influenced by unobserved factors like student motivation or past exposure to the subject.

Second, the skimming phase is defined using curve-based slope criteria determined from aggregated data. Although this is a reproducible and ethical method, it might oversimplify each student's cognitive changes. While some may start answering questions while still in the "skimming" window, others may continue to employ skimming as a method of problem-solving.

Third, even though the linear models and ANCOVA included total test time as a covariate, it might not adequately reflect the complex relationships between time management and skimming. For example, a student who skims efficiently in less time could get the same benefits as one who skims longer (up to 26% of their total time), but such patterns are not explicitly modeled here.

Lastly, the breadth of generalizability is constrained by the major focus on students with low ability. While the study justifies the decision based on previous findings (e.g., Figure 2), it is unknown to what extent skimming improves performance among high-ability students or in different types of questions and the question difficulty. To determine whether the proposed technique offers benefits that are universal or context-specific, future research should examine these dynamics across larger populations and testing situations.

VI. Bibliography

[1] R. Satav, A. Reese, and B. P. Railing, "Statistical Modeling and Analysis of Electronic Examination Logs," Carnegie Mellon University, Pittsburgh, PA, USA, Poster presented at SIGCSE 2025.

[2] GitHub Repository: B. P. Rail, Exam-Time: Tools for Analyzing Exam Behavior, GitHub. [Online]. Available: https://github.com/bprail/exam-time

[3] A. Zoumpoulaki, A. Alsufyani, M. Filetti, M. Brammer, and H. Bowman, “Latency as a region contrast: Measuring ERP latency differences with Dynamic Time Warping,” Psychophysiology, vol. 55, no. 11, pp. 1–13, Sep. 2015. [Online]. Available: https://doi.org/10.1111/psyp.12521

Additional References

B. P. Railing, “Exam Time: How Students Spend Their Time When Taking Exams,” in Proc. 53rd ACM Tech. Symp. Comput. Sci. Educ. (SIGCSE), vol. 2, pp. 1138, Mar. 2022. [Online]. Available: https://doi.org/10.1145/3478432.3499123

VII. Appendix

A. Sensitivity to Skim, Solve, and Review Thresholds

I performed a sensitivity study by modifying the threshold used to determine the transition points between phases in order to guarantee the stability of my Skim-Solve-Review framework. I examined additional thresholds at 20% and 30% to confirm that the results were stable across reasonable fluctuation, even though the main analysis was predicated on the 25% threshold of the score-over-time curve's first derivative (as covered in the Methods section).

Appendix A-1: *ANCOVA Output for Low Ability Students Using Skim Ratio Threshold = 0.2 (Controlling for Total Time)*

```
Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       4.608689   0.160408  28.731  < 2e-16 ***
skim_groupLowSkim -0.530045   0.145757  -3.637 0.000322 ***
total_time        0.001922   0.000737   2.608 0.009526 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.022 on 322 degrees of freedom
Multiple R-squared:  0.1409,    Adjusted R-squared:  0.1356
F-statistic: 26.41 on 2 and 322 DF,  p-value: 2.396e-11
```

Appendix A-2: *ANCOVA Output for Low Ability Students Using Skim Ratio Threshold = 0.3 (Controlling for Total Time)*

```
Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       4.5099116  0.1547952  29.135  < 2e-16 ***
skim_groupLowSkim -0.6128707  0.1459318  -4.200 3.35e-05 ***
total_time        0.0022918  0.0007505   3.054  0.00242 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.071 on 372 degrees of freedom
Multiple R-squared:  0.1698,    Adjusted R-squared:  0.1654
F-statistic: 38.05 on 2 and 372 DF,  p-value: 9.229e-16
```

The ANCOVA results were similar even when the threshold was changed: estimates for low-skim students were approximately -0.5 (-0.53 at the 20% threshold and -0.61 at the 30% threshold), with all models producing a significantly negative coefficient. This shows that low-skim students scored on average 0.5–0.6 points worse than high-skim students, even after adjusting for total time spent. These results support the idea that, regardless of exam length, inadequate skimming is linked to worse performance.

B. Behavioral Trends Among High-Ability Students

Two competing hypotheses were tested to find out why high-ability students typically spend less time solving questions: *H1 that high scorers have more prior knowledge and, as a result, solve questions more quickly and accurately regardless of their skimming behavior; and H2 that high scorers use strategic skimming to think through the problem and cut down on solving time.*

Appendix B-1: *Welch Two-Sample T-Test Comparing Total Time Between Low- and High-Ability Students*

```
        Welch Two Sample t-test

data:  total_time by as.factor(ability_group)
t = 5.3498, df = 630.16, p-value = 1.234e-07
alternative hypothesis: true difference in means between group 1 and group 3 is not equal to 0
95 percent confidence interval:
 23.19459 50.09801
sample estimates:
mean in group 1 mean in group 3
      120.04972       83.40341
```

According to a Welch two-sample t-test (Appendix B-1), high-ability students (group 3) finished the test considerably faster than low-ability students (group 1), average 83.40 seconds as opposed to 120.05 seconds, a difference of 37 seconds that was highly statistically significant ($p = 1.234e^{-07}$). This proves that high scores typically finish faster overall, supporting H1.

Appendix B-2: *Pearson Correlation Between Skim Time and Total Time Among High-Ability Students*

```
[1] "Pearson Correlation: Skim vs Total Time (High Ability)"

        Pearson's product-moment correlation

data:  high_only$Skim and high_only$total_time
t = 70.017, df = 280, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.9654908 0.9782775
sample estimates:
      cor
0.9726104
```

I conducted a Pearson correlation test among the high-ability group to see if strategic skimming plays a role in this efficiency (Appendix B-2). Students who skim more also take longer exams overall, according to the very high and positive association between the amount of time spent skimming and the total amount of exam time ($r = 0.973$, $p < 2.2e\text{-}16$). As skimming seems to add time rather than replace solving problems, this trend defies H2, suggesting that skimming is not a time-saving tactic for these students.

Appendix B-3: *Z-Score-Based Classification of High-Ability Students by Strategy Type*

Strategy	n	Percent
Other	50	17.7
Prior Knowledge	200	70.9
Strategic Skim	32	11.3

To distinguish between previous knowledge and strategic skimming as the dominating behavior, we used a z-score analysis of high scorers' skimming and total time relative to the entire population (Appendix B-3). Most high-ability students (70.9%) meet the "prior knowledge" profile, which is defined by faster completion and below-average skimming. The "strategic skim" tendency, which involves spending more time skimming but finishing on track with average test takers, was only displayed by 11.3% of respondents. The majority of top scorers probably relied more on material familiarity than on time management strategies, which is supported by this classification.

Despite the analysis's finding that prior knowledge is the primary reason why high scores perform better, there is a crucial remark. High scorers spent a larger percentage of their exam time skimming than students with lower ability, as seen in Figure 6 of the Results section. Although this does not imply that skimming increases productivity, it does pose a significant query: could skimming be a useful tactic to boost performance among students with low ability instead? This served as the theme for examining whether skimming, particularly for the low-ability group, increases efficiency.

C. Codebase and Reproducibility Details

Appendix C-1: *Accuracy - Time Data Structuring for Midterm and Final*

Python

```
mid2 <- read.delim("midterm-time.tsv", header = FALSE, sep = "\t")
final2 <- read.delim("final-time.tsv", header = FALSE, sep = "\t")
colnames(mid2) <- c("curr_time","id","question","score")
colnames(final2) <- c("curr_time","id","question","score")
reversed_mid2 <- mid2[nrow(mid2):1, ]
rev_accuracy<- reversed_mid2 %>%
           group_by(id) %>%
           distinct(question, .keep_all = TRUE)
accuracy_mid <- rev_accuracy[nrow(rev_accuracy):1, ]

reversed_final2 <- final2[nrow(final2):1, ]
rev_accuracy_f<- reversed_final2 %>%
           group_by(id) %>%
           distinct(question, .keep_all = TRUE)
accuracy_final <- rev_accuracy_f[nrow(rev_accuracy_f):1, ]


accuracy_mid  <- accuracy_mid  %>%
  separate(question, into = c("category", "version"), sep = "_") %>%
  mutate(category = sub("^q", "", category))

accuracy_final <- accuracy_final %>%
  separate(question, into = c("category", "version"), sep = "_") %>%
  mutate(category = sub("^q", "", category))
accuracy_final$category <- as.numeric(accuracy_final$category)
accuracy_final$version <- as.numeric(accuracy_final$version)
accuracy_mid$category <- as.numeric(accuracy_mid$category)
accuracy_mid$version <- as.numeric(accuracy_mid$version)

viewCount_mid<- mid2 %>% group_by(id) %>% summarise(count = n(),.groups =
"drop")
viewCount_final <- final2 %>% group_by(id) %>% summarise(count = n(),.groups =
"drop")
mid_score_dat <- accuracy_mid %>% group_by(id) %>% summarise(score =
sum(score)/7,.groups = "drop")
final_score_dat <- accuracy_final %>% group_by(id) %>% summarise(score =
sum(score)/7,.groups = "drop")
combined_mid <- left_join(viewCount_mid, mid_score_dat, by = "id")
combined_final <- left_join(viewCount_final, final_score_dat, by = "id")
```

Appendix C-2: *Time Analysis with Respect to Ability Score*

Python

```
mid <- read.delim("midterm-time.tsv", header = FALSE, sep = "\t")
final <- read.delim("final-time.tsv", header = FALSE, sep = "\t")
colnames(mid) <- c("curr_time","id","question","score")
colnames(final) <- c("curr_time","id","question","score")
#Clean the data, removing corrupted log
mid <- mid %>%
  group_by(id) %>%
  filter(all(curr_time >= 0)) %>%
  ungroup()
#Create a new column: the amount of time the student looks at the current
question before switching
mid <- mid %>%
  group_by(id) %>%
  arrange(curr_time) %>%
  mutate(
    time_diff = curr_time - lag(curr_time, default = 0)
  ) %>%
  ungroup()
#The overall time that student take in the exam
mid_summary <- mid %>%
  group_by(id, question) %>%
  summarise(total_time = sum(time_diff)) %>%
  ungroup()

#Clean the data, since midterm made up of 7 questions, if we see entries<7,
then something wrong happens.
mid_summary <- mid_summary %>%
  group_by(id) %>%
  filter(n()>6) %>%
  ungroup()
#Repeat for final exam, but that the final exam made up of 8 question, so the
cleanning data part is different
final <- final %>%
  group_by(id) %>%
  filter(all(curr_time >= 0)) %>%
  ungroup()

final <- final %>%
  group_by(id) %>%
  arrange(curr_time) %>%
  mutate(
    time_diff = curr_time - lag(curr_time, default = 0)
```

```
  ) %>%
  ungroup()

final_summary <- final %>%
  group_by(id, question) %>%
  summarise(total_time = sum(time_diff)) %>%
  ungroup()
#Clean the data
final_summary <- final_summary %>%
  group_by(id) %>%
  filter(n()>7) %>%
  ungroup()


mid_overall <- mid_summary %>%
  group_by(id) %>%
  summarise(total_time = sum(total_time)) %>%
  ungroup()

final_overall <- final_summary %>%
  group_by(id) %>%
  summarise(total_time = sum(total_time)) %>%
  ungroup()

mid_sorted <- mid %>%
  arrange(id, curr_time)
final_sorted <- final %>%
  arrange(id, curr_time)
```

Appendix C-3: *Filtered Final Exam Data Analysis*

Python

```
reversed_final <- final[nrow(final):1, ]
rev_accuracy_fin<- reversed_final %>%
           group_by(id) %>%
           distinct(question, .keep_all = TRUE)
accuracy_final <- rev_accuracy_fin[nrow(rev_accuracy_fin):1, ]

accuracy_final$Roundscore <- ceiling(accuracy_final$score*100 - 0.5)

accuracy_final<- accuracy_final %>%
           group_by(question) %>%
           mutate(Rank = dense_rank(Roundscore)) %>% #Rank of the score
           ungroup()

wide_final <- accuracy_final %>%
  select(id, question, Rank) %>%
  pivot_wider(
    id_cols = id,
    names_from = question,
    values_from = Rank
  )

matrixx <- wide_final[, -1]
library(mirt)
#matrixx  <- matrixx[, !(names(matrixx) %in% c("q20_22", "q15_18"))]
# Fit the GPCM
model <- mirt(matrixx, 1, itemtype = "gpcm")

abilities_finbase <- fscores(model, method = "EAP")

#Second Attempt, first categorize the score inside 10% category(1-10)
#Suppose you have 10~19.9999/100, then you get category 1, 20~29.999 you get
category 2, etc.
accuracy_final <- accuracy_final %>%
                       mutate(Catscore = cut(score,
                       breaks = seq(0, 1, by = 0.1),
                       include.lowest = TRUE,
                       right = FALSE,
                       labels = 1:10))


accuracy_final<- accuracy_final %>%
           group_by(question) %>%
           mutate(Rank_Cat = dense_rank(Catscore)) %>% #Rank of the score
```

```
          ungroup()

wide_final_Cat <- accuracy_final %>%
  select(id, question,Rank_Cat) %>%
  pivot_wider(
    id_cols = id,
    names_from = question,
    values_from = Rank_Cat
  )

matrixx <- wide_final_Cat[, -1]
library(mirt)
#matrixx  <- matrixx[, !(names(matrixx) %in% c("q20_22", "q15_18"))]
# Fit the GPCM
model_Cat <- mirt(matrixx, 1, itemtype = "gpcm")

abilities_fin_Cat <- fscores(model_Cat, method = "EAP")
fin_base <- coef(model, IRTpars = TRUE, simplify = TRUE)
fin_Cat <- coef(model_Cat, IRTpars = TRUE, simplify = TRUE)

joined_final <- inner_join(abilities_final, final_overall, by = "id")
```

Appendix C-4: *Code for Plot in Figure 5: Score Over Time by Ability Group*

Python

```
library(dplyr)
library(ggplot2)

# Convert curr_time from ms to seconds
final2 <- final %>%
  mutate(curr_time = curr_time / 1000)

# Compute time differences (per row, local)
final2 <- final2 %>%
  arrange(id, curr_time) %>%
  group_by(id) %>%
  mutate(time_diff = curr_time - lag(curr_time)) %>%
  ungroup()

# Filter to meaningful time differences only (≥ 3s and < 10.64s)
filtered_df <- final2 %>%
  filter(!is.na(time_diff), time_diff >= 3, time_diff < 10.64)

# Figure 5: Cumulative Score and Time Progression
cumulative_df <- filtered_df %>%
  arrange(id, curr_time) %>%
  group_by(id) %>%
  mutate(
    cum_time = cumsum(time_diff),
    cum_score = cumsum(score),
    total_time = sum(time_diff, na.rm = TRUE),
    total_score = sum(score, na.rm = TRUE),
    cum_time_pct = cum_time / total_time,
    cum_score_pct = cum_score / total_score
  ) %>%
  ungroup() %>%
  filter(
    is.finite(cum_time_pct), !is.na(cum_time_pct),
    is.finite(cum_score_pct), !is.na(cum_score_pct)
  )

cumulative_plot_df <- cumulative_df %>%
  left_join(
    abilities_final %>% mutate(ability_group = ntile(ability_cat, 3)),
    by = "id"
  )
```

```
figure4 <- ggplot(cumulative_plot_df, aes(x = cum_time_pct, y = cum_score_pct,
color = as.factor(ability_group))) +
  geom_point(alpha = 0.2, size = 1.1) +
  geom_smooth(method = "loess", se = FALSE, linewidth = 1.2) +
  labs(
    title = "Cumulative Score Progress vs Time (Per Student)",
    x = "Cumulative % of Total Test Time",
    y = "Cumulative % of Total Score",
    color = "Ability Group"
  ) +
  theme_minimal()

print(figure4)
```

Appendix C-5: *Code for Figures 6, 7, 8, 9 and 10*

Python

```
# Compute mean curve
mean_curve <- per_question_plot_df %>%
  group_by(ability_group, norm_time_q = round(norm_time_q, 2)) %>%
  summarize(mean_score = mean(norm_score_q, na.rm = TRUE), .groups = "drop")

# Fit GAM and compute slope
fit_slope_df <- mean_curve %>%
  group_by(ability_group) %>%
  arrange(norm_time_q) %>%
  do({
    model <- gam(mean_score ~ s(norm_time_q, bs = "cs"), data = .)
    x_vals <- seq(0, 1, by = 0.01)
    preds <- predict(model, newdata = data.frame(norm_time_q = x_vals), type =
"response")
    slopes <- predict(model, newdata = data.frame(norm_time_q = x_vals), type =
"lpmatrix") %*% coef(model)
    data.frame(norm_time_q = x_vals, fitted_score = preds, slope = slopes)
  }) %>%
  ungroup()

# Assign phase labels
get_phase <- function(time, bounds) {
  if (time < bounds$skim_end) return("Skim")
  else if (time < bounds$review_start) return("Solve")
  else return("Review")
}

phase_df <- per_question_plot_df %>%
  left_join(phase_bounds, by = "ability_group") %>%
  rowwise() %>%
  mutate(phase = get_phase(norm_time_q, cur_data())) %>%
  ungroup()

# Compute Time Allocation per Phase
time_allocation <- phase_df %>%
  group_by(id, phase) %>%
  summarize(phase_time = sum(time_diff, na.rm = TRUE), .groups = "drop") %>%
  group_by(id) %>%
  mutate(total_time = sum(phase_time), phase_pct = phase_time / total_time) %>%
  left_join(abilities_final %>% mutate(ability_group = ntile(ability_cat, 3)),
by = "id")

# Figure 6: Average % Time Spent per Phase by Group
```

```
avg_time_pct_by_group <- time_allocation %>%
  group_by(ability_group, phase) %>%
  summarize(avg_pct = mean(phase_pct, na.rm = TRUE), .groups = "drop") %>%
  pivot_wider(names_from = phase, values_from = avg_pct)

print(avg_time_pct_by_group)

# Figure 7: ANOVA by Phase
anova_results <- time_allocation %>%
  group_by(phase) %>%
  do({
    model <- aov(phase_pct ~ as.factor(ability_group), data = .)
    data.frame(phase = unique(.$phase), p_value =
summary(model)[[1]][["Pr(>F)"]][1])
  })

print(anova_results)

# Figure 8: Boxplot
ggplot(time_allocation, aes(x = as.factor(ability_group), y = phase_pct, fill =
phase)) +
  geom_boxplot() +
  facet_wrap(~phase) +
  labs(
    title = "Time Allocation by Phase and Ability Group",
    x = "Ability Group",
    y = "% of Time Spent"
  ) +
  theme_minimal()

# Figure 9: Regression: Phase % ~ Ability
time_allocation_wide <- time_allocation %>%
  select(id, phase, phase_pct) %>%
  pivot_wider(names_from = phase, values_from = phase_pct)

time_allocation_wide <- left_join(time_allocation_wide, abilities_final, by =
"id")

lm_model <- lm(ability_base ~ Review + Skim + Solve, data =
time_allocation_wide)
summary(lm_model)

# Figure 10: Plot relationship (example: Solve vs Ability)
ggplot(time_allocation_wide, aes(x = Solve, y = ability_base)) +
```

```
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm") +
  labs(title = "Solve Time vs Ability", x = "% Time Solving", y = "Ability
Score")
```

Appendix C-6: *Code for Figures 11 and 12*

Python

```
library(dplyr)

# Get latest score per question per student
score_df <- filtered_df %>%
  arrange(id, question, curr_time) %>%
  group_by(id, question) %>%
  summarize(latest_score = last(score), .groups = "drop") %>%
  group_by(id) %>%
  summarize(score_total = sum(latest_score, na.rm = TRUE), .groups = "drop")

# Join score_total into phase_times
phase_times <- phase_times %>%
  left_join(score_df, by = "id")

# Subset Low-Ability Students
low_only <- phase_times %>% filter(ability_group == 1)  # group 1 = Low

# Figure 11: Correlation Test: Skim vs Score
cor_result <- cor.test(low_only$Skim, low_only$score_total, method = "pearson")
cat("\n[1] Correlation: Skim Time vs Total Score (Low Ability)\n")
print(cor_result)

# Figure 12: ANCOVA (Regression Controlling for Total Time)
model_ancova <- lm(score_total ~ skim_group + total_time, data = low_only)
cat("\n[3] ANCOVA: Score ~ Skim Group + Total Time (Low Ability)\n")
print(summary(model_ancova))
```

Appendix C-7: *Code for Figure 14*

Python

```
library(ggplot2)
library(dplyr)
library(mgcv)

# Filter Low-Ability Students and Add Ratio
low_data <- phase_times_02 %>%
  filter(ability_group == 1) %>%
  mutate(
    Skim = as.numeric(Skim),
    total_time = as.numeric(total_time),
    score_total = as.numeric(score_total),
    Skim_ratio = Skim / total_time
  ) %>%
  drop_na(Skim_ratio, score_total)

# Figure 14-1
gam_model <- gam(score_total ~ s(Skim_ratio), data = low_data)
summary(gam_model)

skim_ratio_range <- seq(0.01, 0.9, length.out = 100)  # Avoid 0 or 1
predict_df <- data.frame(Skim_ratio = skim_ratio_range)
predict_df$pred_score <- predict(gam_model, newdata = predict_df)

optimal_row <- predict_df[which.max(predict_df$pred_score), ]
optimal_ratio <- optimal_row$Skim_ratio
cat(sprintf("\n[Optimal Skim Ratio] %.2f (i.e., %.1f%% of total time)\n",
            optimal_ratio, optimal_ratio * 100))

# Figure 14-2
ggplot(predict_df, aes(x = Skim_ratio, y = pred_score)) +
  geom_line(color = "darkgreen", linewidth = 1.2) +
  geom_vline(xintercept = optimal_ratio, linetype = "dashed", color = "red") +
  labs(
    title = "Predicted Score vs Skim Time Ratio (Low Ability Students)",
    subtitle = sprintf("Optimal Skim ≈ %.1f%% of Total Time", optimal_ratio *
100),
    x = "Skim / Total Time (Ratio)",
    y = "Predicted Total Score"
  ) +
  theme_minimal()
```