Comparison of black-box, glass-box and open-box software for aiding conceptual understanding

Conference or Workshop Item

COMPARISON OF BLACK-BOX, GLASS-BOX AND OPEN-BOX SOFTWARE FOR AIDING CONCEPTUAL UNDERSTANDING

Anesa Hosein, James Aczel, Doug Clow and John T.E. Richardson

The Open University

Three mathematical software types: black-box (no steps shown), glass-box (steps shown) and open-box (interactive steps) were used by 32 students to solve conceptual and procedural tasks on the computer via remote observation. Comparison of the three software types suggests that there is no difference in the scores that students receive for conceptual understanding tasks. Students using the black-box are more likely to explore answers than students using the glass and open-box software.

INTRODUCTION

Various mathematical software types such as spreadsheets, CAS or graphic calculators are used at the undergraduate level. These types of software usually function as a black-box (Buchberger, 1990), that is, students input the equations or numbers and through an execute command they receive the answers without seeing the intermediate steps. Whilst the black-box has been applauded in easing the anxiety of weak mathematical students and allowing students to use complex problems, there is concern whether black-box software is the most appropriate tool for students since they are unaware of the processes and have to accept the outputted (Heid & Edwards, 2001) .Buchberger (1990; 2002) suggests that it may be appropriate for some students to use glass-box software which enables the students to see each step before the answer is produced. There is a third type of software that students may use and referred to in this paper as open-box software. Open-box software is where students are able to interact at each step during the solving of the software until the answer is determined. Figure 1 illustrates the three types.

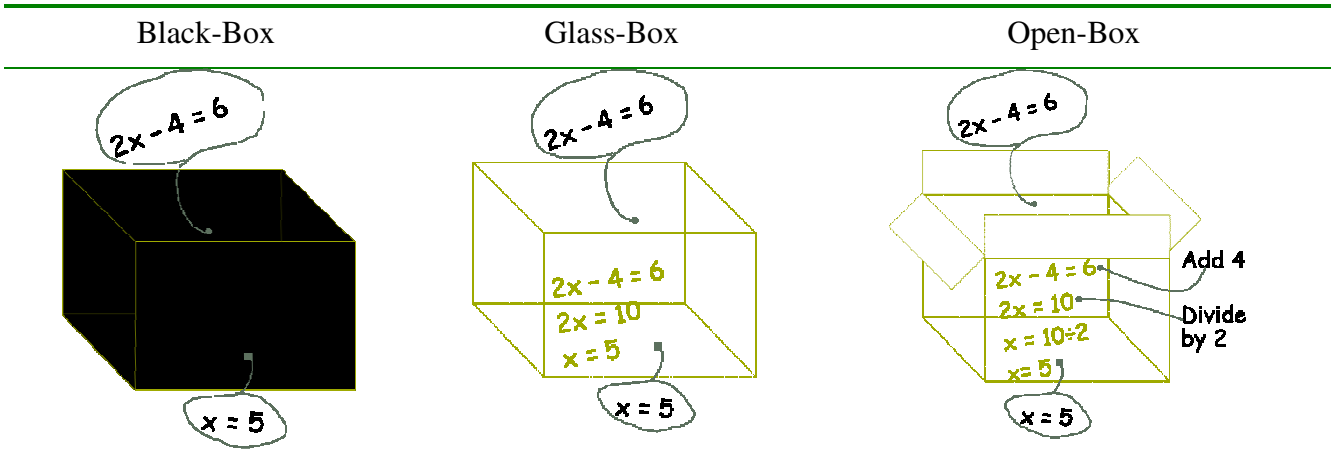

Figure 1: Comparison of an algebra solution by black-box, glass-box and open-box

2008. In Figueras, O. & Sepúlveda, A. (Eds.). *Proceedings of the Joint Meeting of the 32nd Conference of the International Group for the Psychology of Mathematics Education*, and the XX North American Chapter Vol. 1, pp. XXX-YYY. Morelia, Michoacán, México: PME.

There are limited studies in the comparison of the three software types. For example, Horton, Storm, & Leonard (2004) compared the Texas Instruments TI-83 (black-box) and the Casio FX (glass-box) graphing calculators. In their study, over a three week period college students were given problems to solve either in a TI-83 or a Casio FX calculator. At the end of three weeks, students were given a pen-and-paper test and they found that students who used the Casio FX outperformed the students using the TI-83. However, their study looked at only the improvement on mechanical or procedural skills and gave no indication whether the software helped in conceptual understanding. Further, their study measured symbolic manipulation by hand, and whilst this is important, at the tertiary level students are often required to solve problems using software or calculators and such workings have become trivialised.

Perhaps, it may be more appropriate to compare and determine whether these three types of software may have additional advantages over each other such as improving conceptual understanding. Thus, this study investigates how the three software types influences the mathematical understanding of students.

METHODOLOGY

Whilst Horton et al. (2004) only investigated the mechanical or procedural understanding of the students, this research goes further to investigate whether there is any improvement in their conceptual understanding. Thus, a mathematical question taxonomy used by Galbraith & Haines (2000) was employed. They identified three questions types: mechanical, interpretive and constructive. Mechanical questions are mostly related to procedural knowledge, interpretive questions mostly to conceptual knowledge, and constructive questions a mixture of both conceptual and procedural knowledge. Three problems were developed in the linear programming domain which had three parts relating to each of these question types (see Table 1). Linear programming was chosen since a complex problem was needed that students were not familiar with at the tertiary level and could not be easily solved by hand. All mechanical questions were required to be solving used the software. The interpretive questions required the student to either examine or interpret the solution or the problem. The constructive questions had two parts, the first part required the student to use mostly procedural skills to find a different solution for the problem and the second part to use mostly conceptual knowledge to indicate why the different solution worked. All constructive questions were designed to allow the students to solve the procedural part either by using the software or by pen/paper that is through the examination of the problem.

Finding a similar software that displayed all three software types characteristics for linear programming (or for any other mathematical problem) was unsuccessful. Thus, the simplex algorithm used in linear programming was programmed in MS Excel using Visual Basic Application (VBA) to mimic the characteristics of the black-box, glass-box and open-box. As the simplex algorithm involves several choices during an iteration (for example choosing a pivot variable, determining the ratio, choosing pivot

row), the students using the open-box were only required to determine the pivot variable for each step. For all software, the students were aware of when the problem was solved as a pop-up box will indicate that the best solution was found.

Linear Programming Problem:

a) Solve

Max $2x_1 + x_2$

$2x_1 + x_2 \leq 100$ (constraint A)

$x_1 + x_2 \leq 80$ (constraint B)

$x_1 \leq 40$ (constraint C) (**Mechanical**)

b) If x_1 = no. of toy trains manufactured and x_2 refers to the no. of toy soldiers manufactured, and constraint A refers to painting hours, constraint B to carpentry hours and constraint C, the demand for toy trains. Interpret what this solution means to the toy company who wants to maximize their profit by producing toy trains and toy soldiers. Provide as detail answer as possible. (**Interpretive**)

c) If the profit of trains has increased by £1, how would this affect the number of toy trains and toy soldiers being sold? Provide as detail as an answer as possible. (**Constructive**)

Table 1: Illustration of a linear programming problem with the three question types

Data was collected for 36 university students in the UK and Trinidad and Tobago. Students were observed in individual sessions using remote observation (see Hosein, Aczel, Clow, & Richardson, 2007). In the remote observation method data is collected via the internet where students connect to the researcher's computer and uses software on the researcher's computer through application sharing (Figure 2) thus making it practical for collecting data from these two countries.

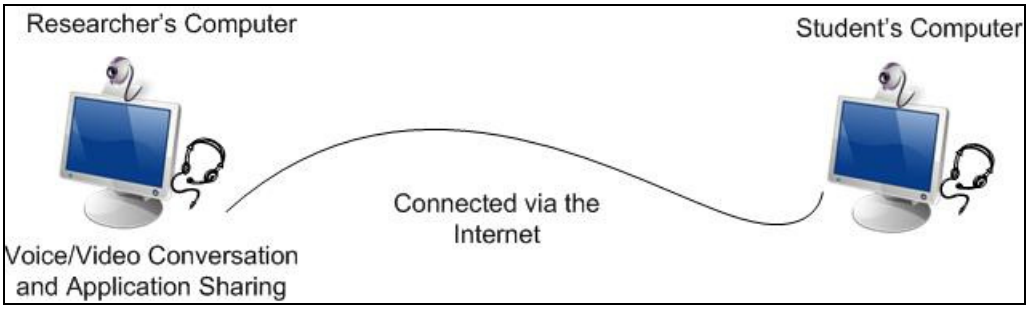

Figure 2: Remote observation process

The observation session was modified from the quasi-experimental framework of Renkl (1997) and Große & Renkl (2006) by adding on the approaches to study inventory (see Table 2). This method was chosen in order to collect both quantitative and qualitative data to allow triangulation. Further, it ensured that data from the

software and the questions types could be partitioned to determine if there were any significant differences.

Students were randomly assigned to use one of the software types to answer all three problems within a Latin square design. Quantitative data was collected from the background questionnaire, pre-test, post-test and the approaches to study inventory. During Step 4, the experiment, students were able to practice with their randomly assigned software and then proceeded to do the three linear programming problems. Their answers were typed and recorded in an answer sheet created in MS Excel. Whilst solving these three problems, students were encouraged to think aloud (Ericsson & Simon, 1984). The think-aloud protocol was used to elicit what self-explanations students were using (Chi, Bassok, Lewis, Reimann, & Glaser, 1989). Students use of the software and their working environment were video recorded from the application sharing process and webcams respectively.

Steps	Instructions
1.Background Questionnaire	Students are asked to fill in a demographic questionnaire, including questions asking for mathematical level, age and gender
2. Study Materials	Students peruse materials to understand the fundamental concepts required for the learning of the topic
3. Pre-test	Students to determine what extent they have prior knowledge of the topic before the stimulus is provided for the experiment. The pre-test problems is at a lower difficulty level than the post-test problems
4. Experiment	Students are provided with the interventions/ factors that are being studied (the type of software)
5. Post-test	Students work on a set of questions to acquire quantitative data to compare the investigated interventions/ factors
6. ASI	Students filled in an approaches to study inventory (ASI) to determine whether a surface or deep approach was used.

Table 2: Modified Quasi-Experimental Method

This paper presents the post-test results for 32 students, 11 using black-box and glass-box each and 10 using the open-box software. For each of the three problems, the students were scored 1 mark for the mechanical part and 2 marks each for the interpretive and constructive part. In this paper, the explanations that students typed for the interpretive and constructive parts were coded into whether the students were relating their explanations to real-life applications and/ or mathematical knowledge. These explanations were part of the students' think-aloud self-explanations. The coding chosen was used to help determine how students were linking their knowledge.

RESULTS AND DISCUSSION

Post-Test Total Mean Scores

The mean scores for each of the software are presented in Table 3. Using an ANOVA, it was found that there was no significant difference in the mean scores from the three software types. All the students achieved full marks for the mechanical part of the problem as was expected since all the students had to use the software to solve the problem. Thus, if there was any significant difference this would have been to the mean scores relating to conceptual understanding. These results perhaps suggest that the three boxes may not improve the conceptual understanding of the students differently.

Software Type	Mechanical	Interpretive	Constructive	Total
Black-Box (11)	3.00	2.96	1.73	7.68
Glass-Box (11)	3.00	2.82	0.95	6.77
Open-Box (10)	3.00	2.95	0.85	6.80
Mean (32)	3.00	2.91	1.19	7.09

Table 3: Score means for the types of questions for the three software types

Students received an average score of 48.5% for interpretive tasks and 19.8% on the constructive tasks. Further examining the constructive tasks, if the constructive tasks were partitioned into its two parts, the students who were able to calculate the procedural part were approximately 30% likely to give a reasonable conceptual explanation for why the procedural part worked (Table 4).

Software Type	Constructive (Procedural)	Constructive (Conceptual)	Constructive (Total)
Black-Box (11)	1.32	0.41	1.73
Glass-Box (11)	0.77	0.18	0.95
Open-Box (10)	0.65	0.20	0.85
Mean (32)	0.92	0.27	1.19

Table 4: Score means for parts 1 and 2 of the constructive problems for the three software types

Further, from an ANOVA, the means suggest that there may be a weak association ($p<0.1$) between the software types and the procedural part of the problem. That is, students using the black-box software appeared to receive scores almost twice those of the students using the glass-box and open-box software in the procedural part of the constructive problem. Whiteman & Nygren (2000) have suggested that black-box software types are useful tools for exploration: that is, for students inputting values and looking at trends. Perhaps students who used the black-box software for exploration in the procedural section of the constructive problems were able to do better. As such, the video data was examined to determine whether students explored using the software for the constructive problem. Those students who explored were coded “yes” for exploration and “no” for no exploration. Although a chi-square

suggests that there was no significant difference in the frequency of exploration for the constructive question by software, the data suggests that students using the black-box (73%) and the glass-box software (64%) had a higher frequency of exploring the constructive task than the open-box (40%).

Further, looking at how students did on the constructive problem on whether they explored or did not explored regardless of the software, it was found that students who did explored, significantly outperformed ($p<0.01$) students who did not explore (1.76 vs 0.35, see Table 5).

Constructive Explored	Constructive (Procedural)	Constructive (Conceptual)	Constructive Score	Total Score
No	0.35	0.00	0.35	6.23
Yes	1.31	0.45	1.76	7.68
Mean	0.92	0.27	1.19	7.09

Table 5: Mean scores for the constructive questions depending on whether the students explored using the software

Further, only students who were able to explore using software to determine the procedural part (unlike those with pen-and-paper) were able to provide a reasonable conceptual explanation. These results imply that although students were able to solve the procedural part either by hand or software, those who did get it correct were more likely to use the software rather than by hand. Further, there was no guarantee that if the students used the software to explore that they were able to obtain the procedural answer, as the average percentage score was approximately 44%.

Explanations of problems

Perhaps further light can be shed on why students did poorly if the explanations can be examined. Coding the explanations from the interpretive and constructive tasks into real-life explanations and mathematical explanations, the results indicate that the students use mathematical and real-life explanations almost equally (Table 6).

	Mathematical Explanations	Real-Life Explanations	Total Explanations
Black-Box (11)	2.6	1.9	4.5
Glass-Box (11)	1.4	2.3	3.7
Open-Box (10)	2.5	1.6	4.1
Total (32)	2.2	1.9	4.1

Table 6: Mean number of explanations that students use for each software box

An ANOVA indicated that there was no significant difference in the mean number of explanations that students used depending on the software, although examining the data there seems to be less mathematical explanations from students using the glass-box software. Examining the conceptual explanations provided for the constructive tasks, there is a clearer indication why students were doing badly in this problem.

There were two main reasons, firstly that students who related their explanations to real-life tended to ignore the underlying mathematics as it relates to the problem (see Table 7). Further, students who used mathematical explanations were sometimes bad at algebra such as understanding the difference between a variable and a coefficient.

"*If the profit per train increased, this means the price of the train increased, if the price of the train is higher than the price of the soldiers, consumers would more likely purchase the cheaper item*" (Glass Box: Real-Life Explanation – ignoring underlying mathematics that $x \leq 40$)
"*Increase profit by £1 may chance constraint C to x<= 40 +1 and since x=40 was our previous answer this may mean it would now mean x increases and y decreases*" (Open Box: Mathematical Explanation – students is changing the right hand side of the equation rather than the coefficient)
*"Profit would increase to 140 but the numbers of toys made stays the same because constraints is that x =40 maximum so even though they get more profit they cant make any more trains" (*Black-Box: Mathematical Explanation – correct explanation*)*

Table 7: Examples of real-life and mathematical explanations made by students for the constructive problem

Also, for the constructive problems, the students who explored using software were significantly likely ($p<0.05$) to give a mathematical explanation than those who did not (1.7 vs 0.9). Students who used real-life explanation gave a similar number of self-explanations whether they explored with software or did not (1.2 vs 0.9). Examining further to determine whether there is any influence from the software types, the type of explanations given for the constructive problems seem to be weakly associated with software type ($p<0.1$). Students using the black-box (1.73) and the open-box (1.5) had a higher mean number of mathematical explanations than that from the students using the glass-box software (0.91). A simple correlation between the scores made for the constructive problem and the types of explanations made found that mathematical explanations positively correlated ($r = 0.62$, $p<0.01$) with the mean constructive whilst the real-life explanations were negatively correlated ($r = -0.37$, $p<0.05$) with the mean constructive scores. This suggests that students who understood the problem mathematically were able to perform better and possibly is dependent on the software.

CONCLUSION

This paper has shown that students using any of the three software types can receive the same mean scores in problems associated with conceptual understanding. However, students using the black-box software are probably more likely to explore numbers and solutions and this may be due to its nature in allowing students to quickly get an answer. Further, students using the black-box and open-box were more likely to give mathematical explanations to conceptual problems than the glass-box

which ensured that they did better overall. Whilst mathematical explanations were expected to be frequent in the open-box and glass box as steps are shown, perhaps the reasoning for the glass-box software having low mathematical self-explanations may be due to the mathematical ability of the students which would have to be further explored.

References

Buchberger, B. (1990). Should students learn integration rules? *ACM SIGSAM Bulletin, 24*(1), 10-17.

Buchberger, B. (2002). Computer algebra: the end of mathematics? *ACM SIGSAM Bulletin, 36*(1), 16-19.

Chi, M. T. H. C., Bassok, M., Lewis, M. W., Reimann, P., & Glaser, R. (1989). Self-explanations: how students study and use examples in learning to solve problems. *Cognitive Science, 13*(2), 145-182.

Ericsson, K. A., & Simon, H. A. (1984). *Protocol Analysis: Verbal Reports as Data*. Cambridge, MA: MIT Press.

Galbraith, P., & Haines, C. (2000). Conceptual mis(understandings) of beginning undergraduates. *International Journal of Mathematical Education in Science and Technology, 31*(5), 651-678.

Große, C. S., & Renkl, A. (2006). Effects of multiple solution methods in mathematics learning. *Learning and Instruction, 16*(2), 122-138.

Heid, M. K., & Edwards, M. T. (2001). Computer algebra systems: revolution or retrofit for today's mathematics classrooms? *Theory Into Practice, 40*(2), 128-136.

Horton, R. M., Storm, J., & Leonard, W. H. (2004). The graphing calculator as an aid to teaching algebra. *Contemporary Issues in Technology and Teacher Education, 4*(2), 152-162.

Hosein, A., Aczel, J., Clow, D., & Richardson, J. T. E. (2007). An illustration of student's engagement with mathematical software using remote observation. In J.-H. Woo, H.-C. Lew, K.-S. Park & D.-Y. Seo (Eds.), *Proceedings of the 31st annual conference of the International Group for the Psychology of Mathematics Education (PME 31)* (Vol. 3, pp. 49-56). Seoul, Korea.

Renkl, A. (1997). Learning from worked-out examples: A study on individual differences. *Cognitive Science, 21*(1), 1-29.

Whiteman, W. E. C., & Nygren, K. P. C. (2000). Achieving the right balance: properly integrating mathematical software into engineering education. *Journal of Engineering Education, 89*(3), 331-336.