Converting an Experimental Laboratory Course From Paper and Pencil to Computer

Maryellen Hamilton
Saint Peter's College

Lisa Geraci
Washington University

This article provides suggestions for developing a laboratory-based research methods course using computers. We describe important considerations for creating this type of course including selecting software, choosing experiments, and teaching students with different levels of computer skill. We also include 3 model projects that required increasing methodological and programming knowledge. The ultimate goal of the course was to have students submit a final project that they designed, ran, and programmed independently.

It has been our experience that most undergraduate students find their experimental laboratory course to be among the most demanding courses they take as part of their psychology major. Traditionally, professors teach this course in a lecture classroom, with students conducting experiments using paper and pencil. However, given that many experiments (particularly those on human cognition) are conducted on computer, it is important to train students to use computers to conduct research. Others have discussed the possibility of enhancing research methodology courses through the use of computers (e.g., Goolkasian, 1985; Peden, 1987; Rittle, 1990); however, this article provides a specific template for creating such a course including experiment and software suggestions. We derived this template from our experience developing and teaching a computer-based research methods course in human cognition at the State University of New York at Stony Brook. Although the class we developed was a research methods course in human cognition, most of the suggestions are applicable to a general laboratory course on research methods.

We describe three main considerations for developing a computer-based experimental laboratory course: selecting software, choosing experiments, and teaching students with different levels of computer skill. We provide solutions that attempt to enrich students' learning of research methods. Our goal for the course was to have students design and program their own independent project on the computer. We required students to program and conduct a series of three experiments prior to completing an independent project. Each experiment increased in theoretical and methodological complexity as well as in programming difficulty. For each successive experiment, students wrote additional sections of an American Psychological Association (APA)-style paper. By the third project they wrote an entire paper.

General Course Considerations

There were three main considerations in developing our computer lab course. First, we had to make software and hardware decisions. We needed experimental software, a program for data analysis, and a way to make stimuli available to each student. We chose Cedrus's Superlab (Abboud & Sugar, 1997) for our experimental software because the program is easy to learn. Superlab is a Windows®-based program that allows experimenters to present stimuli (pictures or words) on a computer screen and record participant responses. The program records accuracy and reaction time for each response and stores the data in a file that students can open using a variety of spreadsheet programs or statistical packages. For example, using a spreadsheet such as Microsoft Excel®, students can sort data, obtain means, and perform data analyses. Students used a private network drive accessible with a password to make changes to their experiment and to continue working on it in subsequent classes. Their network space also gave them a place to save their work for grading. In addition, the entire class could access a network drive where we placed all the materials for each experiment. Although the students had access to this drive, they could not edit it.

Second, we had to select experiments that would work in a classroom setting. Conducting experiments in the classroom required that our students act as both the experimenter and participant in a single study. This decision meant that we had to select experiments with effects that would be obtained even when participants knew of the purpose of the experiment. Also, we had fewer than 20 students in each lab section. Therefore, we had to select experiments with effects large enough to be obtained with a small sample size.

Third, we had to consider that students varied considerably in experience using computers. Some of our students had only word processing experience, whereas others were computer science majors. Therefore, we structured the course around this diversity in computer experience. To aid in this endeavor, the second author wrote a manual specifically for the course.[1] The manual provided novice users with

[1]We encourage professors who want to teach this class to design a manual that is specific to their course needs (e.g., student's abilities, software specifications, course emphasis).

detailed instruction, without forfeiting class time for the more experienced users. Using this manual, students with little computer knowledge were typically able to program simple experiments within a few classes. Another way we dealt with the variability in computer experience was to have students work in pairs. Although students had a computer to themselves, partners could help each other solve programming issues. We rotated partners across each of the three projects so that everyone had an opportunity to work with a more sophisticated computer user.

Specific Course Projects

We selected projects with programming requirements that allowed us to explore specific methodological issues. These selections were important because our goal was to teach research methods as well as computer skills. We used programming specifications to teach students about experimental precision and to make explicit critical research design decisions. For example, certain critical points in programming required students to think about whether the design they were using was within or between subjects. Students needed to make choices about how many lists to program for counterbalancing. The program also required them to code the stimulus presentations (e.g., as studied vs. nonstudied or picture vs. word), which forced them to declare their independent variables and learn about the levels of a variable. We highlighted (both in lecture and in the manual) these issues and presented various programming options to encourage students to think about these design decisions as they programmed. We detail three projects that increase in methodological, theoretical, and programming complexity and required increasingly sophisticated data analyses.

For their first experiment, we chose the classic Stroop (1935) experiment because it met the criteria of having a large effect that was relatively immune to participants' (our students) knowledge of the hypothesis being tested. In addition, this experiment has a simple design with one independent variable with three levels (Stroop, match, and neutral) and one dependent variable (reaction time). Using this experiment as a model, we introduced the following research concepts: hypothesis testing, independent and dependent variables, within- versus between-subject designs, counterbalancing, and the use of ANOVA with post hoc analyses. This experiment required minimal programming because we preprogrammed all the conditions except for the simplest condition, the match condition (where the word matches the color in which it is printed). Each student received a copy of our manual that included a step-by-step procedure for how to program the match condition. The computer skills introduced for this first experiment were creating text stimuli, timing stimuli, randomizing trials, and manipulating and analyzing data using Excel. Students submitted an APA-style report including a title page, abstract, method section, table, and printout of the data analysis.

For the second project, students attempted to replicate a divided attention experiment with ambiguous figures (Reisberg, 1983). The experiment examines why people first see one interpretation and then the other when looking at an ambiguous figure. Whereas in the previous experiment there was only one predicted hypothesis, this experiment had three competing hypotheses regarding the role of attention in mediating this effect. Students tested these hypotheses by comparing the amount of time it took for ambiguous figures to change from the first to the second interpretation under both divided and full attention conditions. This experiment met the criteria previously outlined (i.e., a large effect that could occur with informed participants) and introduced the following: competing hypotheses, confounds, outliers, and the use of *t* tests. It also required minimal programming, which was important because it was the first experiment that students programmed on their own. The design of the experiment was simple and few stimuli were needed. We placed two sets of ambiguous figures (provided in Reisberg, 1983) on a network drive. Half of the class used one set and the other half used the other set so that they were not tested on the same stimuli that they programmed. Students presented one stimulus under divided attention and the other under full attention and created different versions of the experiment to counterbalance the items and the order of conditions. We introduced the following computer skills: creating a new experimental file, accessing stimuli on the network, ending trials with a key press to record reaction time, and creating an instruction screen. For this project each student submitted an APA-style report that included a title page, abstract, method, and results section with data analysis and figure.

For the third project, students conducted a conceptual replication of Weldon and Roediger's (1987) experiment, which demonstrated that the picture superiority effect (better memory for pictures vs. words) can be reversed depending on the cues at retrieval. This experiment had a more complicated design than the previous experiments, two independent variables, and predicted a crossover interaction. Participants studied pictures and words and took either an explicit word fragment cued recall test or a recognition test. Students programmed this entire experiment themselves. So that no one would be tested on the same material that they had spent time programming, we provided students with two different sets of materials on the network drive. We introduced the following research concepts in the third experiment: 2×2 mixed factorial design, counterbalancing for study status and encoding format, main effects and interactions, informed consent, and debriefing. Computer skills introduced in this project included creating study and test files; creating a fixed order for events; coding trials (picture vs. word, studied vs. nonstudied) and coding responses as either correct or incorrect; and sorting data by hits, misses, false alarms, and correct rejections. Unlike the previous two projects, we provided the students with a copy of the Weldon and Roediger's paper to use as a model because this was the first time students wrote an introduction to their APA-style paper. Students submitted a full APA-style report including title page, abstract, introduction, method, results, and discussion, with data analysis and figure.

Conclusions

For the final project, each student independently designed and programmed an original experiment. They gathered their stimuli and created their consent and debriefing forms. Stu-

dents tested a minimum of 20 participants and performed their data analysis. The final result was a full APA-style report and a conference-style presentation with overheads. Forty-six of the 48 students (across three sections) completed the projects successfully. Student projects varied greatly in design and programming complexity. Several of the programs were extremely complicated (including audio), and many students won school research awards for their work in this class.

Students reported on the evaluations for the class that they had learned a great deal in the course and would recommend the course to others. On a scale from 1 (*strong agreement*) to 7 (*strong disagreement*), the students' mean overall satisfaction with the course was 1.81 ($SD = 1.04$), which was better than the university average of 2.08. When asked how they agreed with the statement, "They would highly recommend this course to others" the mean was 2.14 ($SD = 1.01$). In addition, many of the evaluations included comments that the students "learned more in this course than in any other course they had taken in college."

By introducing programming to this course, we believe students were better prepared for graduate work in psychology because they could use computers to implement experiments and analyze their data. We also believe that, by adding the programming component, students thought critically about research design issues (i.e., the experiment will not run appropriately if the design is not precisely specified) and learned to conduct extremely careful experiments.

References

Abboud, H., & Sugar, D. (1997). SuperLab: Experimental laboratory software [Computer software]. Phoenix, AZ: Cedrus.

Goolkasian, P. (1985). A microcomputer-based lab for psychology instruction. *Teaching of Psychology, 12,* 223–225.

Peden, B. F. (1987). Learning about microcomputers and research. *Teaching of Psychology, 14,* 217–219.

Reisberg, D. (1983). General mental resources and perceptual judgments. *Journal of Experimental Psychology: Human Perception and Performance, 9,* 966–979.

Rittle, R. H. (1990). Computer literacy in the psychology curriculum: Teaching a database language for control of experiments. *Teaching of Psychology, 17,* 127–129.

Stroop, J. R. (1935). Studies of interference in serial verbal reactions. *Journal of Experimental Psychology, 18,* 643–662.

Weldon, M. S., & Roediger, H. L. (1987). Altering retrieval demands reverses the picture superiority effect. *Memory and Cognition, 15,* 269–280.

Notes

1. A preliminary version of this article was presented at the seventh annual APS Institute on the Teaching of Psychology, Miami Beach, FL, June 2000.
2. Send correspondence to Maryellen Hamilton, Department of Psychology, St. Peter's College, 2641 Kennedy Boulevard, Jersey City, NJ 07306; e-mail: mhamilton@spc.edu.

Are Online Study Questions Beneficial?

Kristin Grimstad
Jamestown College

Mark Grabe
University of North Dakota

Brothen and Wambach (2001) found that under certain conditions students do not benefit from online resources. They awarded points toward the course grade for high performance on small sets of practice questions and concluded student strategies in completing these quizzes were not beneficial to course performance. We investigated student use of online questions when students received no credit for answering practice questions and found students who made voluntary use of online questions scored higher on course examinations. In contrast to situations in which students complete assigned tasks for credit, students who respond to questions as a voluntary study tactic may use question feedback quite differently and make decisions about how many questions to review using different criteria.

Brothen and Wambach's (2001) study asked whether college instructors should direct students to the online sample quizzes made available by textbook companies. Their study was prompted by survey data (Marek, Griggs, & Christopher, 1999; Weiten, Deguara, Rehmke, & Sewell, 1999) indicating that students used sample test questions rather than other study resources because of their perception that reviewing questions would be a quick way to learn material. Brothen and Wambach argued that to the detriment of course performance students use a "quiz to learn" rather than a "prepare–gather feedback–restudy" strategy when using online questions (Brothen & Wambach, 2001, p. 292).

Brothen and Wambach (2001) offered an online quiz system as part of a course using a modified form of the personalized system of instruction (Keller, 1968). A total of 260 of the 460 examination points available in the course were based on online practice quizzes. Only scores higher than 8 of 10 ques-