ED 259 718 IR 011 751

AUTHOR	Charney, Davida; Reder, Lynne
TITLE	The Role of Examples and Explanations in Teaching Procedural Skills.
SPONS AGENCY	Office of Naval Research, Washington, D.C. Psychological Sciences Div.
PUB DATE	Mar 85
CONTRACT	N00014-84-K-0063
NOTE	20p.; Paper presented at the College Composition and Communications Conference (Minneapolis, MN, March 1985).
PUB TYPE	Viewpoints (120) -- Reports - Research/Technical (143) -- Speeches/Conference Papers (150)
EDRS PRICE	MF01/PC01 Plus Postage.
DESCRIPTORS	Comparative Analysis; Computers; Educational Strategies; *Guides; *Instructional Design; Instructional Improvement; *Intermode Differences; Learning Strategies; *Skill Development; *Teaching Methods; *Textbooks
IDENTIFIERS	*Instructional Effectiveness

ABSTRACT

This paper compares the content of two types of instruction presented to a student either by an intelligent tutoring system or by some conventional text, such as a textbook or a computer user's manual, when the educational goal is skills learning. Two distinct points of view are presented: (1) that of the "expounders," who believe that instruction should be as complete and explicit as possible, and (2) that of the "minimalists," who believe that instruction should above all be brief and should leave much to the learner's own exploration. Each theory is outlined, and the results of a practical application of the expounders theory are reported. In this study, 40 inexperienced computer users were given two manuals to read (one fully elaborated and the other one-third as long with the elaborations deleted), and then asked to perform some tasks on the computer. It was found that learners who had specific tasks in mind when they read the manual performed much more efficiently with the short, unelaborated version of the manual, while unprepared learners did better with the longer version. However, contradictory findings by John Carroll (1984) are noted. He found that after working through his minimal version of a tutorial manual for the IBM Displaywriter System, people learned the same basic information more quickly than those who used the commercially developed version of the manual. This study resolves the contradiction by restating the problem from a new perspective--namely, taking the emphasis away from the length of the instruction and focusing on the inclusion of relevant, essential information, and considering two important dimensions: (1) the kind of information the text must contain; and (2) the kind of learning situation in which it will be used. (JB)

THE ROLE OF EXAMPLES AND EXPLANATIONS IN TEACHING PROCEDURAL SKILLS

Davida Charney
English Department

Lynne Reder
Psychology Department

Carnegie-Mellon University

Paper delivered at the College Composition and Communications Conference, Minneapolis, March 1985.

The work reported here was supported by the Personnel and Training Research Programs, Psychological Sciences Division, of the Office of Naval Research, under Contract No. N00014-84-K-0063, Contract Authority Identification Number, NR667-529.

IR011751

Introduction

This paper concerns the content of the instruction presented to a student either by an intelligent tutoring system or by some conventional text, such as a textbook or a computer user's manual (which is the kind of text we have recently been most involved with). To state the question directly: when the goal is to design instructional materials for skill learning, how much instruction should be provided and just what must it consist of? Our goal in attempting to answer this question is to develop guidelines for designers of educational software and writers of instructional texts, guidelines that reflect cognitive science research into what instruction students need in order to learn a skill.

Writers and instructional designers who are concerned about this issue have tended to fall into two quite distinct camps: the "expounders," who believe that instruction should be as complete and as explicit as possible, and the "minimalists," who believe that instruction should above all be brief and should leave much to the learner's own exploration. The fact is that both points of view have some merit and both are backed up by some experimental research. So the instructional designer is still left in the lurch: when do you expound and when do you minimize? What we plan to do in this paper is first to outline each of these two positions and the support that has been mustered for them. Then we will try to resolve the conflict of the two positions by describing the conditions under which more or less instruction seems to be needed.

The Case for the Expounders

The expounders' view is the more traditional: an instructional text for novice learners should be as complete as possible; it should assume little if any prior knowledge and should provide detailed exposition of all relevant points. This point of view is held by writers of both textbooks and computer manuals. Robert Tausworthe, for example, outlines several levels of detail for documenting computer software. The highest level of detail is called for in what he labels "Class A documentation," which he describes as follows:

> "Class A documentation is the most detailed; it contains specific definitions and detailed descriptions of every significant factor or item within the software specification..... This level of detail probably finds its most applicability in user manuals, and rightly so: The writer of a user manual is generally unavailable for consultation, so the user needs the extra detail." [Tausworthe (1979), pp. 158-159.][1]

Some designers of intelligent tutoring systems hold a similar view: Since students become frustrated by repeated early failures, tutoring systems should be designed to protect students initially from committing too many errors. Designing such systems involves both organizing the material skillfully and presenting more explicit instruction, especially in the early stages.

Recently, we did some experimental studies that produced evidence supporting the expounder's case (Reder, Charney & Morgan, 1984). We prepared two versions of a computer user's manual which taught novices a basic set of commands for the operating system of the IBM Personal Computer. One version of the manual was fully elaborated with definitions, analogies, examples, metastatements about the text's organization, and so on. The second version of the manual was about one-third as long (3500 words as opposed to 11,000), and omitted all of these elaborations. Figure 1 illustrates the differences between the two versions with side-by-side excerpts.

In our experiment, we brought in 40 inexperienced computer users, gave them one of the two manuals to read, and then took away the manual and asked them to perform some tasks on the computer. In addition to varying the amount of elaboration in the manuals, we also designed our experiment to recreate two common learning situations. Sometimes people have specific goals in mind and turn to instructional materials to find information relevant to those goals. At other times, people come to learn a new skill with only a general idea of how they will make use of what they learn. We simulated these two learning conditions by giving half of our subjects advance information about the tasks they were going to perform. These subjects would then read the manual with the specific tasks in mind.

So the basic design of the experiment was as follows. We divided our subjects into two groups. Subjects in the "Before" group read the instructions for the tasks they would be expected to do on the computer before they read the user's manual. Subjects in the "After" group were simply told that the tasks would call on information in the manual; they saw the instructions for the tasks only after they finished reading the manual. Within these two groups, half of the subjects read the elaborated manual and half the unelaborated manual. Subjects were given 45 minutes to read their version of the manual and were told that the manual would not be available to them after the reading period was over. After the subjects read the manual, they were asked to carry out the tasks on the computer: renaming files, creating subdirectories, copying and deleting files, and so on. (The exact list of tasks is found in Figure 2). As the subjects worked, the computer kept a record of every command they typed and the time at which it was entered. The measures of how well subjects performed were whether they were able to do the tasks and how efficiently they worked (i.e., how much time they took and how many commands they had to issue to the computer).

The results showed very different trends for the Before and After groups. The Before group, the subjects who had specific tasks in mind when they read the manual, performed much more efficiently with the short, unelaborated version of the manual. On the other hand, the After group performed much better with the longer, elaborated manual. Figure 3 shows this pattern for the average number of commands subjects issued to complete the tasks.[2]

Even though we found the shorter, unelaborated manual to work better for the Before group, the results in general support the expounder's case. As writers of instructional texts, we can't assume that all learners will come in with such clearly defined goals as the Before group subjects. In fact, the subjects who read the unelaborated manual without having specific goals in mind consistently had the worst performance. On balance, these learners seemed more greatly impeded by under-elaborated texts than the more directed learners were by the over-elaborated version. So it seems advisable to play it safe and provide elaborated instruction to all learners.

Our study, then, provides some support for the traditional view that instruction should be complete and explicit. However, our experiment made one crucial simplifying assumption that does not seem to hold good in the real world. We ensured that our subjects would sit down and read through the manual before they began work on the computer. The fragility of this assumption is where the Minimalists begin their case.

The Case for the Minimalists

Observation of learners actually using instructional texts such as computer manuals shows that they are often quite unwilling to read instructional information, even when the relevant passage is easy to locate. They prefer to figure things out on their own, or to ask someone (cf. Wright, 1983; Scharer, 1984; Carroll, 1984). John Anderson (Pirolli & Anderson, 1984) has observed that even when learners do read the text, they seem to use relatively little of this information during task performance. From this point of view, providing complete and detailed instruction is of little practical use to the learner.

Designers of so-called "minimalist training materials" proceed on the assumption that willingness to read a manual is inversely related to the manual's length, that people in general want to start doing things instead of reading about them and that therefore, instructional materials should actively encourage discovery learning by providing as little prose as possible. As John Carroll (1984) describes it: "The first principle of Minimalist design is to slash the verbiage; that's where the name comes from (i.e., less to read can mean better training)." He put this principle into practice in a tutorial manual for a commercial word processing system (the IBM Displaywriter System) and produced a revised manual that was one-fourth the length of the original. Carroll's principles for shortening the manual included two major steps. First, he slashed everything he considered irrelevant to the task at hand,

"...eliminating all repetition, all summaries, reviews, and practice exercises, the index, and the troubleshooting appendix. ...All material not related to doing office work was eliminated or radically cut down (the welcome to word processing overview, descriptions of the system status line, details on the system components..., etc.)." [p.5]

Carroll's second step was to take what was left, the relevant information, and delete parts that he believed learners would be able to learn on their own.

"Procedural details were deliberately specified incompletely to encourage learners to become more exploratory, and therefore, we hoped, more highly motivated and involved in the learning activity (e.g. the function of the cursor-step-keys was introduced with an invitation to 'Try them and see.')" [Carroll, p.6][3]

Carroll's manual was tutorial in the sense that readers were expected to try things out as they read about them. Carroll found that after working through his minimal version of the manual, people learned the same basic information more quickly than people who used the commercially developed version of the manual. Furthermore, when Carroll's subjects went on to study more advanced topics, they learned more new techniques more quickly if their initial training had been conducted with the minimalist materials. While Carroll himself admits that these initial efforts at designing minimalist materials have been exploratory (e.g., as a result of the testing, he found he had to put back some explanatory sections as well as some procedures that subjects actually couldn't figure out on their own), Carroll's findings in the main support the minimalist position: having less to read leads to equivalent or better learning at a faster overall rate.[4]

Resolving the Contradiction

So here we have two seemingly contradictory positions, each with experimental evidence to back it up. It might seem tempting at this point to conclude that experimental data are useless for resolving writing controversies. However, being made of sterner stuff, we propose to resolve the contradiction by restating the problem from a new perspective. The mistake that both expounders and minimalists often make is to focus on length per se (or degree of detail) as the critical issue. Length is not really the issue. The expounders' goal is not to write the longest text possible: they in fact try to produce texts that are as concise and relevant as possible, given what topics must be covered. In an important respect, the minimalists try to do exactly the same thing. Obviously, the difference lies in what they consider this relevant, essential information to be.

Our position is quite simple: when you teach a skill, you have to convey information of various kinds. Some kinds of information should be elaborated with illustrative examples and details, other kinds do not require such a lengthy treatment.[5] If we are right, then the expounders are including irrelevant information by giving detailed treatment to all points and not just the ones that need it.[6] Conversely, the minimalists may be underspecifying some points when learners would benefit greatly from more elaboration.

Our more recent work and Charney's dissertation research investigate two concerns: (1) in what kinds of learning situations do learners benefit from having more detailed information and (2) what kinds of information need such elaboration? Each of these questions adds an important dimension to the question of instructional content. In this section, we will discuss each of these dimensions, starting with the second: what kinds of information are involved in skill learning and should some receive more elaboration than others?

Kinds of information

We believe that good skill performance on a novel task requires four things:

1. Appreciating the meaning of novel concepts and procedures. For example, in learning to use an IBM Personal Computer, one might be introduced to special function keys that allow the user to edit and re-execute a command that was issued previously, without retyping the command. The learner has to appreciate the concept of reissuing a command, as well as learning the specific functions of each of the special keys.

2. Remembering to use the procedure. "Knowing" at some level that such function

keys exist does not mean that the user will remember to use them at the appropriate time rather than retyping the command.

3. Selecting the most appropriate procedure for the situation at hand. Even if learners do remember the function keys in time, the situation may allow alternative combinations of the keys to be used. At this point, learners have to know enough to choose among the options intelligently.

4. Remembering the exact requirements of the procedure and understanding how to fulfill them in a specific situation. Once the learner has decided to use a procedure, he or she must know how to execute it correctly. For example, where must the cursor be positioned? Is pressing the function key enough or must you also type a carriage return?

Elaborations in the text may touch on any of these topics, the basic concepts, when they are relevant and how one applies them. For example, when the goal of a task does not exactly match the function of any known procedure, a subject with deeper understanding of the function of each individual procedure may more easily construct an effective combination. Elaborations about what conditions affect the usefulness of a procedure can help subjects plan out more efficient sequences of actions. Finally, supplementing a general syntactic rule for a computer command with specific examples can help subjects set more specific standards for what their own commands must look like.

In our first experiment, we used manuals that either elaborated on all of these points or on none of them. However, not all types of information may need the elaboration. We tested this possibility by intuitively classifing the elaborations in our manual into two groups. Elaborations were classified as "conceptual" if they concerned basic concepts, such as the purpose of a command or when to use it. That is, these elaborations dealt with any of the first three types of information described above. Elaborations were classed as "procedural" if they concerned the fourth type of information, how to issue commands correctly (e.g., examples of commands, details about notation conventions, etc.). Four versions of the manual were then produced: one contained both conceptual and procedural elaborations, one contained neither type, one contained just the conceptual elaborations and one just the procedural ones. In this way, we could tell whether the advantage we found for the elaborated manual in our first experiment was due to the conceptual elaborations, the procedural elaborations, or both.

This experiment was conducted in a very similar fashion to the first experiment, except that no subjects were given advance information about the tasks they would perform. So all subjects started by reading one version of the manual. Then the manual was removed, and subjects performed a set of tasks on the computer. This time, our subjects included 40 novice computer users and 40 experienced computer users, none of whom had ever used an IBM-PC. We expected that the novices might need elaborations of both kinds, while the experienced computer users (who were already familiar with basic computer concepts) might only need the procedural elaborations.

We found that regardless of previous computer experience, subjects who had read the manuals containing procedural elaborations were by far the most efficient at task performance. The conceptual elaborations seemed to have no effect whatsoever: having the conceptual elaborations alone produced performance no better than having no elaborations at all, and adding the conceptual elaborations to the procedural elaborations was equivalent to having the procedural elaborations alone. Figure 4 shows this pattern, again for the number of commands subjects issued to complete the tasks.

This experiment provides striking evidence that user's manuals need procedural elaborations: both experienced and novice computer users learn better with them than without them. The finding that the conceptual elaborations were useless even for the novice computer users is a bit surprising. Charney's dissertation research follows up on this result by separating out the various types of information we had lumped together as "conceptual." In particular, she is looking at how people decide when to use one procedure rather than another. Certain kinds of conceptual information may play a large role in this kind of decision and may very well work better with elaboration.

The important conclusion to draw from this study is that we should not try to adopt a uniform level of detail for every aspect of an instructional text. It seems possible to sort out different kinds of information that work best with different degrees of exposition. It would be a mistake, however, to conclude that certain types of information always need a certain level of detail. The studies described above all dealt with a particular kind of skill learning situation, learning to use a computer system. If we consider what is involved in this kind of learning situation, it may become clearer why people benefitted from additional information on how to issue commands but did not benefit from additional conceptual information.

Kinds of learning situations

A learning situation typically involves three things: a learner, some means of instruction (for our purposes, an instructional text), and a task or job that the learner ultimately wants to do. In order to distinguish different kinds of learning situations, it is useful to think about how much the learner must depend on the text in order to do the task. Let's consider a few typical instructional texts:

- Instructions for assembling a bicycle or setting the date on a digital watch.
- Cookbook recipes, directions for knitting a sweater.
- Manuals for using a computer program or operating system.
- Textbooks/intelligent tutoring systems about math, writing, physics or computer programming.

The items on this list are arranged in order from most to least learner dependence of the text. Consider the first situation. If you're reading instructions for putting together a kid's bicycle, you've got all the parts right in the carton, there's only one thing you can make out of those parts and there's only one right way to put the parts together (not counting minor variations like adjusting the seat and pedals, attaching optional training wheels, etc.). You accomplish the task by reading a step in the instructions and then immediately carrying it out. There's no need to "master" the instructions: you don't need to remember them once bike is put together, you don't expect them to help you accomplish any other task. Since you can go back and forth from the instructions to the task, you don't even need to hold the current step in memory very long. So, in this situation the learner is very dependent on the text. The learner consults the text at every point during performance of the task and expects to use the instructions every time the task arises, but not at any other times.

Because of this dependence, the instructions can consist almost exclusively of procedural information: the list of steps to be carried out in sequence. Conceptual information can be omitted altogether, as long as the assumptions of the situation are maintained. This is the result that Smith and Goodman (1982) obtained when they studied people learning to put together simple flashlight circuits. They found that adding conceptual information about the component parts of a circuit (e.g., a circuit consists of a battery, a switch and a light) or information about how a circuit works (e.g., how electricity flows through it to light the bulb), didn't help people assemble the circuit more efficiently. However, when these same people were later asked to diagnose what was wrong with a faulty circuit, those who had learned how a circuit works did the best job. They also did a better job at reassembling the circuit from memory, though this advantage was not significant. The point is that asking people to apply what they know or to work from memory departs from the assumptions of this type of learning situation, so other types of information may become important to have.[7]

Contrast the assembly type of situation with the situation we have been talking most about: learning to use a computer system. The biggest difference is the relationship between the information in the text and what the learner wants to accomplish. In the bicycle situation, you want to put a bicycle together and that's exactly what the instructions tell you how to do. In the computer situation, the procedures you learn about are a means to a more distantly removed end. You turn to a manual because you want be able to write letters or analyze data or plan a budget on the computer. The manual doesn't talk about how to write a letter; it talks about how to start a file, how to enter text, how to move the cursor and how to save the file.

The instructional text, then, is teaching general procedures that you can apply over and over again as part of a whole range of tasks. The goal is to work independently of the text. When you know how to use a computer operating system or a text editor, for example, you have learned the most common commands and procedures so well that you use them at the appropriate time almost without thinking about them.

The features of this learning situation have some important consequences for what the instructional text can look like. For example, the procedural information can't be as specific as in the assembly situation. The assembly instructions for a bicycle can refer directly to the parts you will work with, such as "insert the front wheel assembly sprocket into the lateral widget." The writer of a computer manual has much less certainty about what you are working with or what you want to accomplish. So computer manuals contain procedural information in the form of abstract rules that can cover a large number of situations. As the user, you have to figure out how to apply the rule correctly to your particular task. The only way for the manual to be specific is to supply a hypothetical example or other elaborations about how the rule works.

The amount and type of procedural information needed in the bicycle assembly type of situation is quite different from that needed in a computer manual. So we cannot conclude that certain types of information always need a certain level of detail.

Conclusion

The goal of this talk was to explore decisions about the content of instructional texts: what to say and how much to say. First, we contrasted two overly s[illegible]ws of this issue. In one view, instructional texts for novices should be completely explicit and detailed. In the other, instructional texts should contain nothing but the bare minimum of information. We argued that the problem is more complex than a simple dichotomy between long and short. There are two important dimensions that must be considered: (1) what kind of information the text must contain and (2) what kind of learning situation it will be used in. When both of these dimensions are taken fully into account, we may realize the goal of creating the shortest possible instructional text that also teaches skill most effectively.

References

Carroll, J. (March 1985). *Designing MINIMALIST Training Materials* Research Report 46643). IBM Watson Research Center, Computer Science Department. Also appeared in Datamation 30(18), 125-136, 1984.

Pirolli, P. & Anderson, J. R. (in press). The Role of Learning from Examples in the Acquisition of Recursive Programming ~~Skills~~. *Canadian Journal of Psychology*.

Price, J. (1984). *How to Write a Computer Manual: A Handbook of Software Documentation*. Menlo Park, CA: The Benjamin/Cummings Publishing Co.

Reder, L., Charney, D., & Morgan, K. (August 1984). *The Role of Elaborations in Learning a Skill from an Instructional Text* Technical Report 1). Carnegie-Mellon University.

Reder, L. M. (1985). Techniques available to author, teacher and reader to improve retention of main ideas of a chapter. In Segal, J., Chipman, S., & [illegible]zer, R. (Ed.), *Thinking and learning skills: Current research and open questions*. Hillsdale, N.J.: Erlbaum.

Scharer, L. (July 1983). User Training: Less is More. *Datamation*, 29, 175-182.

Tausworthe, R. (1979). *Standardized Development of Computer Software. Part II.* Englewood Cliffs, NJ: Prentice-Hall.

Wright, P. (December 1983). Manual Dexterity: A User-Oriented Approach to Creating Computer Documentation. In A. Janda (Ed.), *Human Factors in Computing Systems*. Boston: Computer-Human Interaction.

Notes

[1]For a more recent statement of a similar position, see Price (1984).

[2]The difference between the manuals arose in how efficiently subjects worked. Subjects completed about the same number of tasks with either manual. We take this to mean that both manuals were adequate for learning the necessary information, but the elaborations made the information easier to use efficiently

[3]It is worth emphasizing that Carroll was very selective and very deliberate about what information to leave out of his minimal manual. Carroll's "missing information" is therefore quite different from the all too common blunder made in many commercial computer manuals of blithely or inadvertently leaving out crucial steps. The major danger in Carroll's approach is that other, less careful writers might conclude that they have carte blanche to "slash verbiage" at will because people "learn better from shorter manuals." The fact that Carroll is not as indiscriminate as this in practice is an important one, and one that we will return to shortly.

[4]It should be pointed out that Carroll made other changes to the manual in addition to making it shorter: he clarified the terminology and organized the discussion around typical situations for users It is therefore uncertain how much of the superiority of the shorter version is due to length and how much to these other changes.

[5]Charney's dissertation focusses on how to draw this distinction reliably. It could be that the reader's purpose in reading a text is the sole determinant of whether a given piece of information should be elaborated or not. On the other hand, it could be that there are some identifiable types of information, some of which should always receive detailed treatment regardless of the reader's immediate purpose

[6]Reder (1985) discusses the cost to the learner of having to read irrelevant elaborations in textbooks. Irrelevancies distract the reader's attention away from the important information and make the important information harder to recall.

[7]It is interesting to note that the benefit derived from the funtional elaborations -- how the circuit worked, rather than from the structural elaborations -- how the parts related to each other. The benefit derived from a certain type of conceptual understanding, not from conceptual information per se

Figure 1

Samples of four types of elaboration

Meta-statement

ELABORATED

Since the computer has two disk drives which can each contain a diskette, you must specify whether the file you want is in drive A or drive B when you give the computer a command. If your command doesn't specify which drive contains the file, the computer automatically assumes that it can find the file in the "default" drive. The next section explains what the "default drive" is, and how to tell the computer to look on a different drive if necessary.

UNELABORATED

Since the computer has two disk drives which can each contain a diskette, you must specify whether the file you want is in drive A or drive B when you give the computer a command. If your command doesn't specify which drive contains the file, the computer automatically assumes that it can find the file in the "default" drive.

Definition

ELABORATED

The B: ("B-colon") in the command stands for the right-hand disk drive. The colon signals the computer that the letter or word preceding it is a "device" rather than the name of a command or file. Devices are pieces of computer hardware, such as disk drives, a printer or even the keyboard. After you enter the command, the B> prompt will appear on the screen. From now on the computer will automatically look for files on drive B.

UNELABORATED

The B: in the command stands for the right-hand disk drive. From now on, the B> prompt appears on the screen and the computer will automatically look for files on drive B.

Figure 1, continued

Analogy

ELABORATED

When you give the computer a command concerning a file, such as TYPE, ERASE or COPY, the computer looks for the file on a "diskette." A diskette, also known as a "floppy disk," is similar to a small, flexible phonograph record record, except that instead of storing sounds, it contains information which the computer can read, add to or delete. All the files you create on the computer are stored on diskettes. So, in order to work on your files, you must insert the diskette that contains them into the computer. You insert a diskette into one of the two "disk drives" on the front of the computer cabinet. The drive on the left is called drive A, and the one on the right is drive B.

UNELABORATED

When you give the computer a command concerning a file, such as TYPE, ERASE or COPY, thecomputer looks for the file on a "diskette." To use a diskette, you insert it into one of the two "disk drives" on the front of the computer cabinet. The drive on the left is called drive A, and the one on the right is drive B.

Figure 1, continued

Example

The elaborated and unelaborated versions are for the most part identical except for the addition of the example (italicized here)

Using COPY to Combine Files

You can use COPY to combine files, appending a copy of one file to the end of another file.

FORMAT

The format of the command is:

```
COPY   [Loc & name first file+next file+...]  [Loc & name combined file]
```

[Loc & name first file+next file...] refers to a list of the files you want to "add" together. The names of the files are typed with plus (+) signs between them. You need to specify location information for each filename in the list in the usual way, with drive and path specifications. When several filenames are listed in this manner, the COPY command results in a new file in which the contents of the first file on the list appear first, followed by the contents of the second file, then the contents of the third file and so on. So be sure that the files in the list appear in the order in which you want them combined.

[Combined file] refers to the new file that will contain the combined files; what you want to call this file, and where in the directory structure you want it to go. Specify the location in terms of a drive and a path to a directory as usual. Type the name you would like to give the file at the end of the path.

For example, suppose you write a report in sections, with each section in a separate file. You want to format and print the report as one file, so you combine the sections into one file. The following command takes three files, INTRO.MSS, BODY.MSS, and CONCL.MSS and combines them into a new file called REPORT:

```
A> COPY   B:INTRO.MSS+B:BODY.MSS+B:CONCL.MSS   REPORT   <ENTER>
```

The combined file, REPORT will consist of the introduction, the body and the conclusion.

Figure 2

Instructions for Tasks on the IBM-PC

These tasks will allow you to practice using the concepts you learned from the manual. You may work on these tasks in any order. Continue working until you are satisfied that you have completed the tasks to the best of your abilities. We want you, however, to work as efficiently as possible.

Task 1. Before you, in drive A, is a diskette containing a number of files. Some of these files have the word "PART" in their names, such as file "PART.1." We want you to change the names of these files. The new name that you should give each file appears as the first line of that file. So, inspect the contents of each file that now has "PART" in its name and give the file the name that you find on the first line of the file.

Task 2. Four of the files on the diskette have the word "DATA" in their names, and the abbreviation of a month in their extension, such as DATA.MAR. We want you to create a fifth data file named ALLDATA.83 that contains the contents of the other four data files appended together. Within ALLDATA.83, the files should appear in "chronological" order; that is, the contents of DATA.MAR should precede the contents of DATA.JUN because March is earlier in the year than June.

Task 3. Next, you should create two subdirectories, on the diskette in drive B. One subdirectory is to be named PROGRAMS and the other named DATA. Move all the files that have the word "Program" in their names from drive A into the PROGRAMS directory on drive B. And, similarly, move the "Data" files (including ALLDATA.83 from Task 2, if you have already created it), into the DATA directory. You do not want any Program or Data files to remain on the diskette in drive A.

Task 4 Finally, you should eliminate the SOURCE directory and everything it contains from the root directory of the diskette in drive A. The root directory on drive A should now contain only a list of files.

Your task is complete!

Figure 3

The Role of Author-provided Elaborations

(when studying a manual with and without prior knowledge of tasks to be performed)

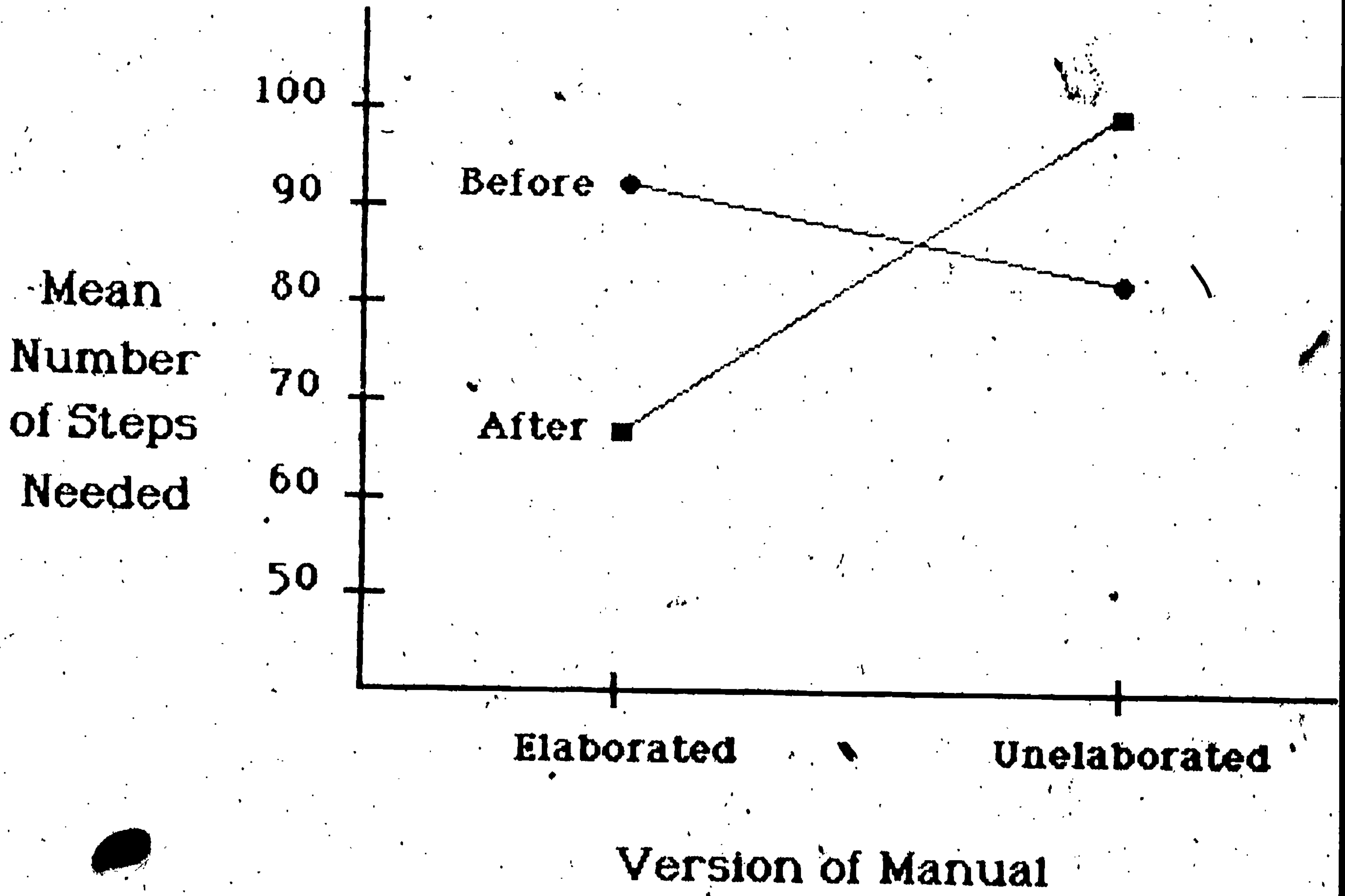
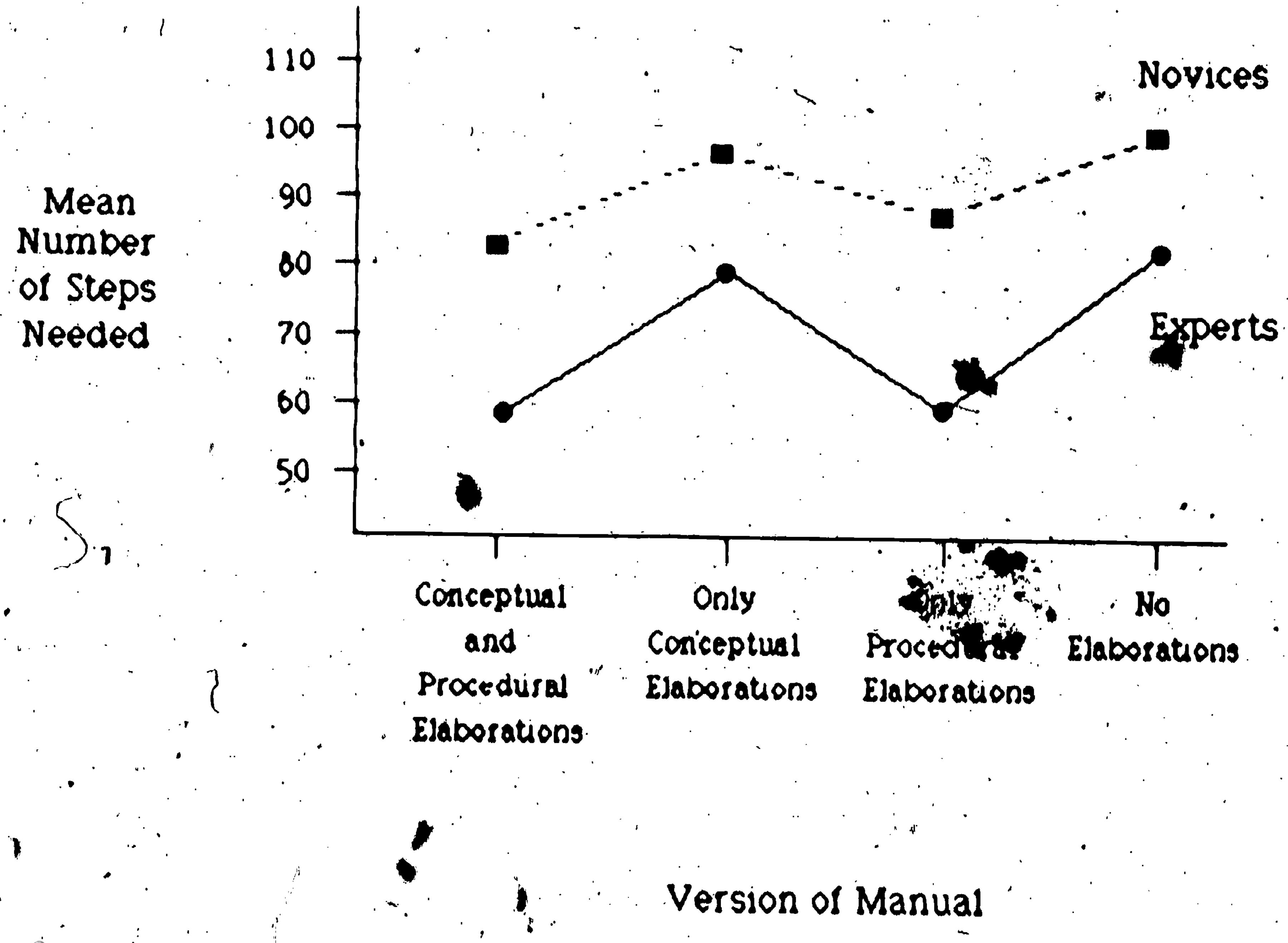


FIGURE 4

The Effect of Elaborations of Different Types on Skill Learning


Mean Number of Steps Needed
Novices
Experts
Conceptual and Procedural Elaborations
Only Conceptual Elaborations
Only Procedural Elaborations
No Elaborations
Version of Manual