

# Distinguishing Qualitatively Different Kinds of Learning Using Log Files and Learning Curves

Kenneth R. Koedinger, Santosh Mathan<sup>1</sup>

Human Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, PA 15213

[koedinger@cmu.edu](mailto:koedinger@cmu.edu), [mathan@alumni.carnegiemellon.edu](mailto:mathan@alumni.carnegiemellon.edu)

**Abstract:** This paper illustrates a technique for determining whether the knowledge acquired by students over the course of an instructional intervention can be used flexibly across problem contexts. The technique relies on power law fits to training data obtained from log files. Power law curves can also provide insight into the learning rate and helps researchers determine when differences among instructional conditions begin to emerge.

**Keywords:** Power Law of Learning, Cognitive Tutors, Transfer

Simply comparing learning outcomes that result from two or more distinct instructional treatments leaves numerous questions of interest to instructional researchers unanswered. Differences between pre-test and post-test scores reveal little about the flexibility with which students will be able to use their newly acquired skills. Learning outcomes do not reveal enough to determine whether the treatment or some other pre-existing characteristic of students in a group contributed to the observed difference between treatments. Even in cases where no apparent difference exists in terms of learning outcomes, the course of the learning process may be of interest to researchers. Student may have reached a particular level of competency relatively early in one condition over the other.

In this paper we describe a technique that employs learning curves derived from logs generated by model tracing tutors to make inferences about the course and effectiveness of learning. We do so in the context of a spreadsheet tutor designed to teach students spreadsheet cell referencing skills. The approach described here requires that the target skill or knowledge be decomposed into component skills or knowledge elements. In Cognitive Tutors (e.g., Corbett, Koedinger, & Hadley, 2001) we use ACT-R (Anderson & Lebiere, 1998) inspired production rules as the knowledge elements, however, the approach we describe is neutral to the particular representation of knowledge components and could be constraints (e.g., Mitrovic & Ohlsson, 1999), difficulty factors (e.g., Heffernan & Koedinger, 1998), or any other knowledge representation method. We call these knowledge elements “learning factors” and the approach described here is an instance of Learning Factors Analysis (Koedinger & Junker, 1999; Heffernan, Croteau, & Koedinger, 2004). The approach requires that the errors associated with each opportunity to practice a skill component be logged. Learning curves generated from such logs can describe performance at the start of training, the rate at which learning occurs, and the flexibility with which the acquired skills can be used.

## Power Law Fits of Data

Newell and Rosenbloom (1993) have observed that skill improvement with practice follows power law, which appears as a linear relationship in log-log space. The greatest improvement on a skill occurs early in the learning process and slows down with subsequent practice (see figures 1 and 2 for examples of power law curves).

The general function for this power law relationship is:  $E = Xn^\alpha$

E describes the error rate, or other performance criteria of interest.

X describes performance on the first trial.

n describes the opportunity to practice a skill

$\alpha$  describes the learning rate

---

<sup>1</sup> Author is currently affiliated with Honeywell Labs, 3660 Technology Dr, Minneapolis, MN 55418

Anderson, Conrad, and Corbett have demonstrated that the power law relationship between practice and performance may not always be readily evident in the practice of complex skills. For instance student performance over the course of individual steps in each practice problem in the LISP tutor showed a rather chaotic pattern. However, by decomposing the skill required to program in LISP in terms of production rules, and examining performance as a function of the opportunity to practice each of the underlying rules Anderson and colleagues observed a clear power law relationship between practice and performance.

Power law fits of learning outcomes as a function of the opportunity to practice basic knowledge elements can provide useful parameters that describe performance on the first learning trial and the learning rate. These parameters characterize the course of the learning process. This paper demonstrates that power law fits of practice data can also shed light on the depth of learning – whether the knowledge acquired can be used flexibly across contexts. We describe this technique in the context of a spreadsheet tutor described below.

### **The Learning Context: A spreadsheet tutor**

Power law fits of training data provided valuable insight into the relative efficacy of two versions of a spreadsheet tutor. Mathan and Koedinger (2003) compared the relative effectiveness of a tutor emphasizing solution generation skills with a tutor that also supported the exercise of error detection and correction skills. The so-called expert model version of their tutor provided feedback on the basis of a model that emphasized error free and efficient task performance. Students were kept on a solution path – deviations from the solution path were remedied with immediate feedback. In contrast, a version of the tutor based on the model of a so-called intelligent novice allowed students to make errors and observe the consequences of errors. The intelligent novice version gave students an opportunity to analyze their errors and generate a new solution. If student's failed to correct errors at the solution step, corrective feedback served to get students back on the solution path.

We hypothesized that the opportunity to reason about errors in the intelligent novice condition would facilitate a deeper conceptual understanding of domain principles. As Merrill, Reiser, Merrill, and Landes (1995) have theorized, errors provide an opportunity to develop a better model of the behavior of operators in a domain. They attribute this to the fact that error recovery requires that students construct explanations about the causes and consequences of errors and act on their analyses. This kind of active self-explanation and problem solving, they argue, contributes to a better understanding of domain operators and their applicability in problem contexts.

Comparisons of the expert and intelligent novice versions of the tutors relied on tests of procedural knowledge and conceptual understanding immediately following training. Our assessments also included transfer and retention tests. Students in the intelligent novice condition outperformed students in the expert condition on all measures. However, even such a broad set of outcome assessments did not serve to clarify whether these differences are qualitative, that is, they emerged as a consequence of the opportunity to build a different, more general model of domain operators as a result of reasoning about errors, or whether the differences are quantitative, that is, they emerged from a more efficient acquisition of the same domain operators. Although randomization of subjects to condition and the statistical results make it unlikely, it is also possible that prior knowledge and experience among students in the intelligent novice condition account for or contributed to the observed difference. Pre-test assessments of competence showed prior knowledge among students in both groups to be close to zero. We turned to an analysis of power law learning curves to rule out two of these three possibilities.

### **Assessing depth of encoding with Power law fits**

Over the course of their interaction with the spreadsheet tutor, students worked with variations of six types of problems represented in the tutor (Table-1) (see appendix 2 for overview of tutorial domain). There are several ways in which a learner may encode the knowledge necessary to solve these problems. A more shallow or superficial encoding would require a unique rule for each type of problem. A deeper, more flexible knowledge encoding, would result in the acquisition of a smaller set of rules that would apply more broadly across problems. For instance, problems one and two in the table below require no absolute references because the formula will refer to cells at the same relative location as the original formula no matter where it is pasted.

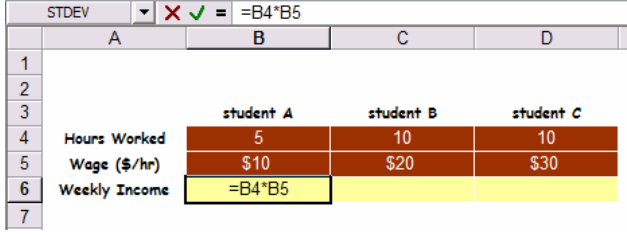
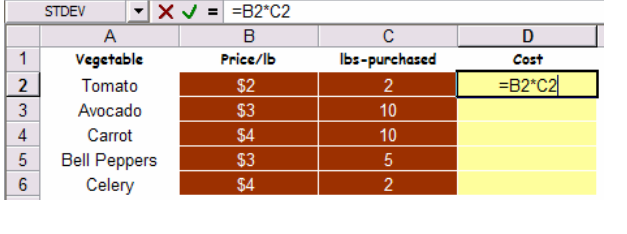
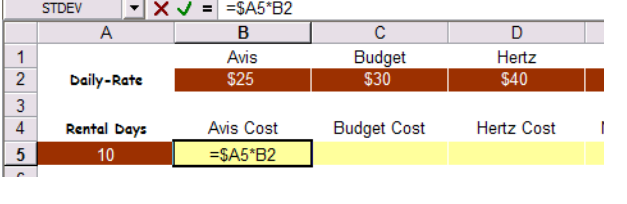
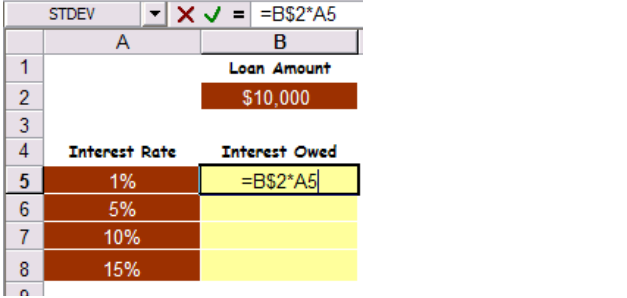
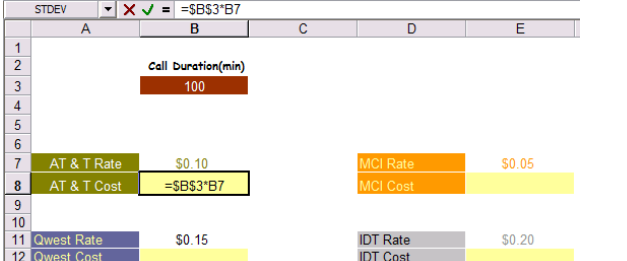
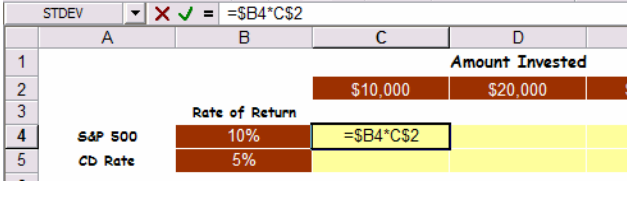
<p>Example of each type of problem represented in the tutor</p>	<p>Shallow encoding: Each problem type has a distinct knowledge component associated with it.</p>	<p>Deep encoding: Certain knowledge components generalize across multiple problem types.</p>
	<p>1. Relative-row</p>	<p>1. Relative</p>
	<p>2. Relative-column</p>	
	<p>3. Absolute-row</p>	
	<p>4. Absolute-column</p>	<p>2. Absolute</p>
	<p>5. Double absolute</p>	<p>3. Double absolute</p>
	<p>6. Matrix</p>	<p>4. Matrix</p>

Table 1: Six types of problems represented in the spreadsheet tutor with shallow and deep skill encodings.

A deep and flexible rule would treat the first two problems the same way. However, a shallow knowledge encoding would rely on superficial features of the problems such as the orientation (in a row vs. in a column) of the cells to be pasted into. This shallow encoding will result in two rules, one relative referencing rule relevant to the horizontal orientation of paste cells, the other to the vertical orientation of paste cells.

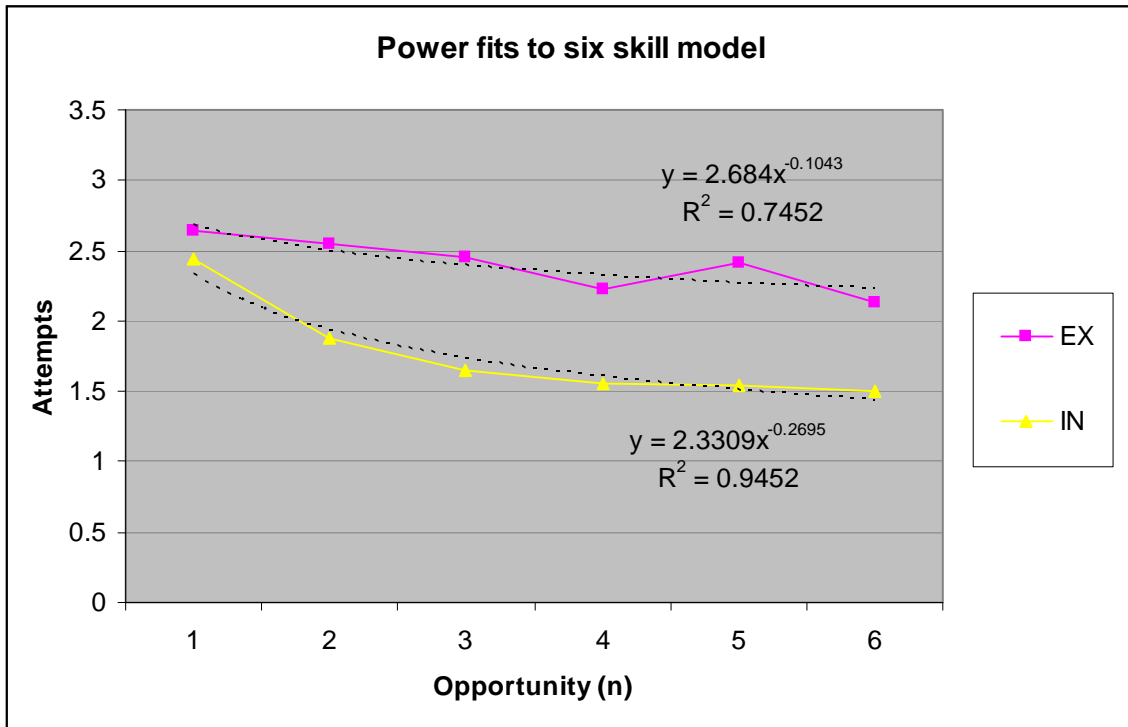
An analysis of learning curves provides the basis for determining whether students acquire a shallow or deep encoding of skills. Consider the two relative referencing problems in the table below. If students were to encode a separate rule for each of the two types of cell referencing problem, each iteration through the six types of problems would provide a single opportunity to practice the two distinct production rules associated with two relative referencing problem types. In contrast, with a deeper encoding, where a single production rule would relate to the two problem types, students would receive two opportunities to practice the general production rule applicable to both problem types. Yet another general encoding one could make with the type of problems highlighted in the table below would be to have a single production rule that applied to both the third and fourth problems.

We created plots of the number of attempts required to generate correct solution to the problems as a function of the opportunity to practice underlying production rules (see figures 1 and 2 below). We created two plots, each with a different assumption about the underlying encoding. We created one plot assuming a six skill, shallow encoding associated with the six types of problems represented in the tutor and another representing a four skill encoding of the six types of problems. Thus, with each iteration through the six types of problems, learning data organized with respect to a six skill encoding would show a single opportunity to apply each production rule. In contrast, with a four skill, deep encoding, the log data is organized such that there are two opportunities to apply the general rule relevant to the two types of relative referencing problems and two opportunities to practice the general rule associated with the two single absolute reference problems. Additionally there would be one opportunity each to apply each of the two rules relevant to the two other types of problems represented in the tutor. Fitting power law curves to data plotted with these alternative assumptions about the underlying skill encoding can help determine whether students were acquiring a skill encoding that would generalize well across problems or not.

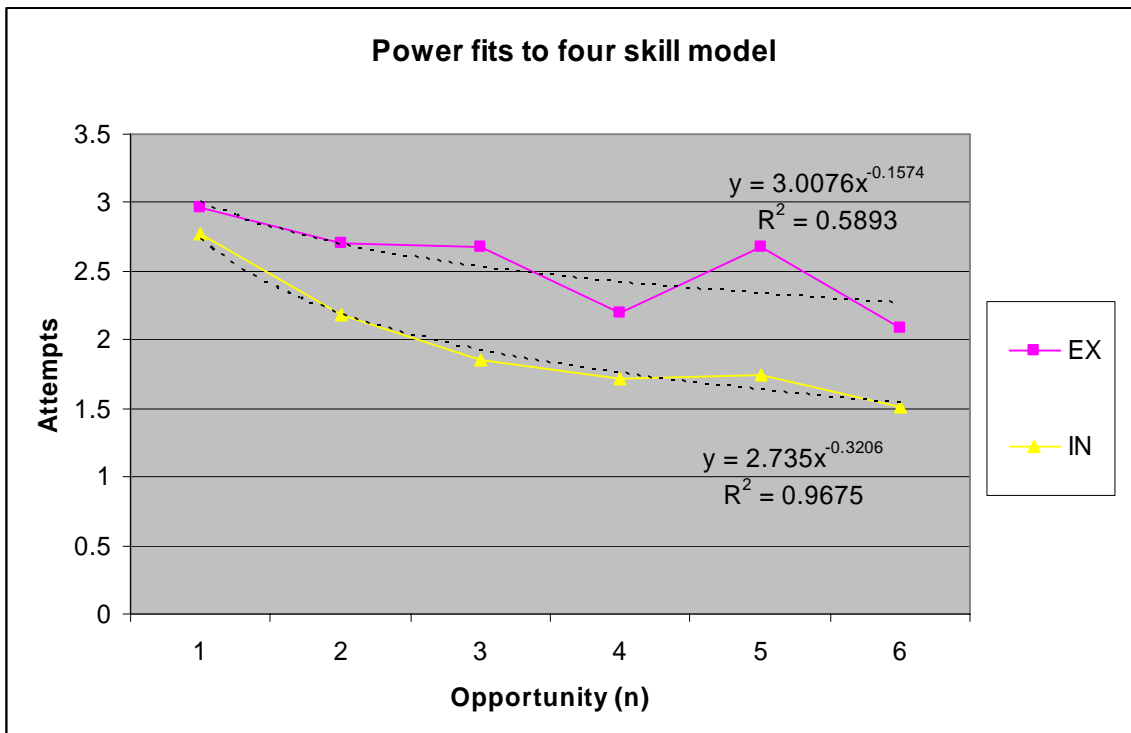
It is worth illustrating how these two different knowledge or skill models, the shallow vs. deep, make different predictions about the shape of the learning curve. Let us consider how students acquiring the shallow six-skill model are likely to perform as they repeat instances of problems in the six categories in Table 1. The shallow six-skill model predicts that the learning experiences on the first six problems will be independent of each other (the  $X$  and  $\alpha$  in the power law formula above), but that performance on the next six problems should improve. That is, the error rate ( $E$  in the formula above) should be smaller on the second opportunity to apply each of these skills (when  $n = 2$  in the formula) than it is on the first opportunity (when  $n = 1$ ). On the third iteration through these six problems ( $n = 3$ ), the error rate will be even lower.

Now, let us contrast the predictions of the four-skill model. For students acquiring the deep four-skill model, the learning experiences on the first six problems will *not* be independent of each other. Instead, the second and fourth problems will be repetitions ( $n = 2$ ) of skill elements (the “relative” and “absolute” skills in the third column of Table X) practiced previously in the first and third problems. Thus, for students acquiring the deep four-skill model, there should be *transfer* of the learning experience from problem 1 to problem 2 (the relative skill) and from problem 3 to problem 4 (the absolute). And thus, performance will increase (the error rate will drop) more quickly on these problems than it will for students that acquire the shallow six-skill model.

Figures 1 and 2 show power law fits of the two instructional conditions, intelligent novice and expert, to the two alternative skill models. The x-axis in the graphs represents the number of opportunities to practice a skill, the “ $n$ ” in the formula. The y-axis represents an alternative version of error rate where instead of computing the proportion incorrect (0 if a step is performed correctly on the first attempt and 1 otherwise), we computed the number of attempts a student made until they finally got the answer correct (1 if a step is performed correctly on the first attempt, 2 if correct on the second attempt, and so forth). The points plotted are the average number of attempts across all students and all skills (6 per point in Figure 1 and 4 per point in Figure 2) for each opportunity number ( $n$ ).



**Figure 1: Power law fits to a more shallow six-skill model. Note how the expert (EX) condition fits better here than in the deeper four-skill model in Figure 2.**



**Figure 2: Power law fits to a deeper four-skill model. Note how the intelligent novice (IN) condition fits better here than in the more shallow six-skill model in Figure 1.**

We performed a power law fit to each of the four sets of data points in Figures 1 and 2. Note that all models provide a reasonable fit to the learning curve data with  $R^2$  values ranging from 0.59 to 0.97. The important result is that the learning behavior of students in the Intelligent Novice condition is better fit by the deeper four-skill model ( $R^2 = 0.97$ ) than the shallow six-skill model ( $R^2 = 0.95$ ) whereas the learning behavior of the students in the Expert condition is better fit by the shallow six-skill ( $R^2 = 0.75$ ) model than the deeper four-skill model ( $R^2 = 0.59$ ). In other words, the learning curves provide evidence that students getting Intelligent Novice instruction acquire a deeper, more flexible encoding of the domain operators whereas the students getting Expert instruction acquire a shallower, less general encoding of the domain operators. These fits demonstrate that students in the intelligent novice condition were acquiring a qualitatively better skill encoding. That is, students in the intelligent novice condition were acquiring skills that would apply broadly across problem types.

These plots also provide some insight into other aspects of the quality of learning fostered by each version of the tutor. From both plots, it is clear that students in both conditions start off at the same level. The advantage for the intelligent novice condition is only evident in the second attempt to practice problems. It is also clear that students in the intelligent novice condition demonstrate a higher learning rate than students in the expert condition. The exponent in the power law function ( $\alpha$ ) represents the learning rate. Power fits associated with both the six skill and the four skill reveal a learning rate in the intelligent novice condition that is twice as high as the expert condition (six skill – EX: -0.10 IN: -0.27; four skill – EX: -0.16 IN: -0.32) (See figures 1 and 2). The appendix steps the reader through the procedure of conducting the analysis described here on the basis of log file data.

## Conclusion

This paper illustrates a technique for determining whether the knowledge acquired by students over the course of an instructional intervention can be used flexibly across problem contexts. The technique relies on power law fits to training data obtained from log files. Analysis of power law curves also provides insight into the learning rate and helps researchers determine when differences among instructional conditions begin to emerge.

While the analysis illustrated here has relied on averaging over students, an alternate approach might be to perform such an analysis based on individual student data. For instance, we could have fit the 4-skill vs. 6-skill models to individual students and then count the number of students in each condition that better fit one vs. the other.

## References

- Anderson, J. R., Cornrad, F., & Corbett, A., (1989) "Skill acquisition and the Lisp Tutor," *Cognitive Science*
- Anderson, JR & Lebiere, C. (1998). *The atomic components of thought*. Mahwah, NJ: Erlbaum.
- Corbett, A. T., Koedinger, K. R., & Hadley, W. H. (2001). *Cognitive Tutors: From the research classroom to all classrooms*. In Goodman, P. S. (Ed.) *Technology Enhanced Learning*. Mahwah, NJ: LEA.
- Heffernan, N., Croteau, E., & Koedinger, K. R., (2004). Why are algebra word problems difficult? Using tutorial log files and the power law of learning to select the best fitting cognitive model. To appear in *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*.
- Heffernan, N. & Koedinger, K. R. (1998). A developmental model for algebra symbolization: The results of a difficulty factors assessment. In Gernsbacher, M. A. & Derry, S. J. (Eds.) *Proceedings of the Twentieth Annual Conference of the Cognitive Science Society*, (pp. 484-489). Mahwah, NJ: Erlbaum.
- Koedinger, K. R. & Junker, B. (1999). *Learning Factors Analysis: Mining student-tutor interactions to optimize instruction*. Social Science Data Infrastructure Conference. New York University. Nov, 1999.
- Mathan, S. and Koedinger, K. *Recasting the feedback debate: Benefits of tutoring error detecting and correction skill*. Int. Conf. Artificial Intell. Education (2003)
- Newell, A, Rosenbloom, P.S. (1993). Mechanisms of skill acquisition and the law of practice. In P. S. Rosenbloom, J. E. Laird, & A. Newell (Eds.), *The Soar Papers: Research on Integrated Intelligence*. The MIT Press.
- Mitrovic, A., & Ohlsson, S. (1999). Evaluation of a constraint-based tutor for a database language. *International Journal of Artificial Intelligence in Education*, 10(3-4), 238-25

## Appendix 1: Analysis Procedure

This appendix outlines the steps we took to go from log file data to the learning curve analysis. Note, this procedure is an instance of Learning Factors Analysis (Koedinger & Junker, 1999; Heffernan, Croteau, Koedinger, 2004). Figure 3 illustrates part of the file from which the Learning Factors Analysis was performed. Columns A-E and J and K come directly from the tutor interaction log data. The Problem Number column displays the order in which the student saw each problem. The “Attempts required to correctly solve” shows how many times the student tried to write a formula before getting it correct. The average of this value, across different problem types and students, is displayed on the y-axis of Figures 1 and 2. The remaining columns show the individual formula attempts the student made.

Columns F-I are created as the key step in producing learning curves. Columns G and I are the knowledge element or skill coding for each of the problems for the six-skill and four-skill models, respectively. The skill names correspond to those in Table 1. The values in column F are the number of opportunities (the N in the power of law of practice formula) the student has had to practice the skill in column G (the six-skill model). Similarly, the values in column H are the number of opportunities the student has had to practice the skill in column I (the four-skill model). Notice how the opportunity counts between the six-skill model (Column F) and four-skill model (Column H) deviate in row 4 of the table. This problem is the third the student has attempted (Problem Number = 3). It is an instance of Problem Type 4 and is encoded in the six-skill model as “relative row” (meaning it is a relative reference, and it is copied within a row). Because this problem is the first opportunity the student has had to apply the “relative row” skill, the “opportunity to practice six-skill knowledge elements” (column F) is 1. In contrast, in the four-skill model, Problem Type 4 is encoded as “relative”. Because the student has applied this skill before (in problem 1), this 3<sup>rd</sup> problem is the second opportunity for the student to apply this skill. Thus, the “opportunity to practice four-skill knowledge elements” (column H) is 2. In other words, the four-skill model predicts transfer between instruction on problem 1 and performance on problem 3 whereas the six-skill model predicts students will be learning anew from problem 3.

The final steps in our analysis involve drawing learning curves and fitting a power law function to the data. Such data analysis can be done with a number of software packages. We have used Microsoft Excel. Excel’s “Pivot Table” feature makes it relatively simple to make an x-y function table relating Opportunity values (e.g., Column F) with the average of the Attempts (column J) across all rows that have the same Opportunity value. For instance, for the six-skill model and Opportunity = 1, the Attempts in rows 2-7 will be averaged together (for Student #1 shown in this excerpt) along with all the other rows (not shown in this excerpt) where Opportunity = 1 for all the other students in the data set. Once this x-y function table is created (relating Opportunity and average Attempts), Excel’s charting features can be used to create a scatterplot of the data. These features also include options for curve fitting and a power function is one of the possible choices.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	StudentID	Walkthrough	Feedback	Prob Num	Prob Type	Opportunity to practice six-skill knowledge elements	Six-skill element name	Opportunity to practice four-skill knowledge elements	four-skill element name	Attempts required to correctly solve	Formulas Entered			
2	1	noWT	IN	1	3	1	relative row	1	relative	1	'=B4*B5			
3	1	noWT	IN	2	2	1	absolute row	1	absolute	2	'=B2*A5	'=B\$2*A5		
4	1	noWT	IN	3	4	1	relative col	2	relative	1	'=B2*C2			
5	1	noWT	IN	4	1	1	absolute col	2	absolute	1	'=\$A5*B2			
6	1	noWT	IN	5	5	1	double abs	1	double abs	3	'=\$B3*B7	'=B\$3*B7	'=\$B\$3*B7	
7	1	noWT	IN	6	6	1	matrix	1	matrix	3	'=B4*C2	'=B\$4*C\$2	'=\$B4*C\$2	
8	1	noWT	IN	7	3	2	relative row	3	relative	1	'=B3*B4			
9	1	noWT	IN	8	2	2	absolute row	3	absolute	1	'=\$B\$1*B4			
10	1	noWT	IN	9	4	2	relative col	4	relative	1	'=\$B2*\$C2			
11	1	noWT	IN	10	1	2	absolute col	4	absolute	2	'=\$A5*B2			
12	1	noWT	IN	11	5	2	double abs	2	double abs	2	'=\$B1*B5	'=\$B\$1*B5		
13	1	noWT	IN	12	6	2	matrix	2	matrix	3	'=\$A5*C2	'=\$A\$5*\$C2	'=\$A5*\$C\$2	
14	1	noWT	IN	13	3	3	relative row	5	relative	1	'=B3*B4			

Figure 3: Part of file on which analysis described on preceding pages is based.

## Appendix 2: Tutorial Domain — Spreadsheet Cell Referencing

A spreadsheet is essentially a collection of cells on a two dimensional grid. Individual cells may be addressed by their column and row indices. Column indices (also called column references) are denoted by letter, whereas row indices (often called row references) are denoted by number. Cells may contain alphanumeric data and formulas. Formulas can refer to values in specific cells by referring to their addresses. So a user could enter a formula in cell C3 (in column C and row 3) that adds the content of cell A3 and B3 by typing “=A3+B3”.

Formulas may be reused to perform iterative operations. This is accomplished through a scheme called relative referencing. Consider the spreadsheet depicted in Figure-4. One could enter a formula in cell B5 that adds the contents of cells B2, B3, and B4. The corresponding operation can be performed in cells C5 and D5 simply by copying the formula entered in cell B5 and pasting it into these new locations. When pasted, Excel modifies the formula to refer to cells that lie at the same relative location as the original formula. For example the formula in Cell B5 referred to the 3 cells above it. When the formula is copied and pasted into cells C5 and D5 the formulas are modified to refer to the three cells above these new locations.

	A	B	C	D	E
1		January	February	March	Total
2	Rent	700	700	700	=B2+C2+D2
3	Electricity	60	53	72	=B3+C3+D3
4	Phone	100	58	75	=B4+C4+D4
5	Total	=B2+B3+B4	=C2+C3+C4	=D2+D3+D4	

	A	B	C	D	E
1		January	February	March	Total
2	Rent	700	700	700	2100
3	Electricity	60	53	72	185
4	Phone	100	58	75	233
5	Total	860	811	847	

**Figure 4: Relative Referencing allows formulas in B5 and E2 to be reused via copy and paste**

In order to determine the appropriate relative references at new locations, Excel updates formulas based on where the formula is moved. When a formula is moved into a cell in a different column, Excel updates column references in the formula by the number of columns moved (see Figure-8, =B2+B3+B4 becomes =D2+D3+D4 when moved across columns from B5 to D5). Similarly, when a formula is copied and pasted into a cell in a different row, all row references in the formula get updated by the number of rows moved (see Figure-8, =B2+C2+D2 becomes =B4+C4+D4 when moved across rows from E2 to E4).

	A	B
1		Hourly Wage
2	Hours Worked	\$10
3	20	=A3*B2
4	30	=A4*B3
5	40	=A5*B4

	A	B
1		Hourly Wage
2	Hours Worked	\$10
3	20	\$200
4	30	6000
5	40	240000

	A	B
1		Hourly Wage
2	Hours Worked	\$10
3	20	=A3*\$B2
4	30	=A4*\$B2
5	40	=A5*\$B2

	A	B
1		Hourly Wage
2	Hours Worked	\$10
3	20	\$200
4	30	\$300
5	40	\$400

**Figure 5: Inappropriately used relative references (left) remedied with absolute references (right)**

While relative referencing works in many task contexts, it is sometimes necessary to hold a row or column reference fixed regardless of where a formula is moved. Consider the example in Figure-5. The value in cell B2 (Hourly Wage) has to be multiplied with the values in cells A3, A4, and A5. If the formula, =A3\*B2 is entered into B3 and pasted into cells B4 and B5, all row references will change in order to refer to cells that lie at the same relative location as those referred to by the formula in B3. This would produce =A4\*B3 in B4 and =A5\*B4 in B5 (instead of =A4\*B2 and =A5\*B2 respectively). In order for the formula to continue to refer to cell B2, the row reference 2 has to be held fixed as an absolute reference. This can be done by placing a ‘\$’ ahead of ‘2’. Thus, in order for the formula in B3 to work appropriately when copied and pasted, it would be modified to read =A3\*\$B2.