Reducing the Knowledge Tracing Space

Steven Ritter[1], Thomas K. Harris[2], Tristan Nixon[1], Daniel Dickison[1], R. Charles Murray[1] and Brendon Towle[1]
[1]Carnegie Learning
[2]EDalytics, LLC

Abstract. In Cognitive Tutors, student skill is represented by estimates of student knowledge on various knowledge components. The estimate for each knowledge component is based on a four-parameter model developed by Corbett and Anderson [Nb]. In this paper, we investigate the nature of the parameter space defined by these four parameters by modeling data from over 8000 students in four Cognitive Tutor courses. We conclude that we can drastically reduce the parameter space used to model students without compromising the behavior of the system. Reduction of the parameter space provides great efficiency gains and also assists us in interpreting specific learning and performance parameters.

1 Introduction

Since their start over 15 years ago, Cognitive Tutors [9] have used Corbett and Anderson's [4] knowledge tracing algorithm as a method for estimating student knowledge. The knowledge tracing algorithm models student understanding as a collection of knowledge components (also called skills). Task performance depends on whether students have the requisite knowledge and whether they are able to exhibit that knowledge within the task. Knowledge components are assumed to be either known or unknown, and the system's task is to estimate the probability that each of the target knowledge components are known. The model uses two knowledge parameters: *pinitial*, the probability that the knowledge component was known prior to instruction within the software; and *plearn*, the probability than an unknown knowledge component will transition to the known state, given an encounter with a task requiring that knowledge. The model also incorporates two performance parameters, which are meant to explain why performance of a task does not exactly match the state of student knowledge. The two performance parameters are *pslip*, the probability that a student will make an error when the knowledge component is known; and *pguess*, the probability that the student will provide the correct answer when the knowledge component is unknown.

At each opportunity to use a skill, *pknown*, the system's estimate of the probability that a particular knowledge component is known, is updated as a Bayesian function of the four parameters (*pinitial* being the initial *pknown*). Since *pknown* at any point is dependent only on the prior *pknown* and the three other knowledge tracing parameters, this model is a variant of a hidden Markov model, and we can use various techniques to estimate the best-fitting parameters for each knowledge component [7].

The benefits of setting knowledge tracing parameters based on student data were empirically demonstrated by Cen et. al. [3], who fit knowledge tracing parameters based on data collected for one cohort of students, and used the new parameter settings within an optimized version of the tutor. Students using the optimized tutor were able to reach mastery in 12% less time, relative to an identical system without the optimized parameters, while maintaining equivalent performance on both immediate and delayed post-tests.

This result demonstrates the value of using educational data to improve the performance and efficiency of Cognitive Tutors. Our goal in this paper is to explore the sensitivity of the Cognitive Tutor to the particular parameters used to model students in order to see if we can reduce the search space of knowledge tracing parameters. In particular, we would like to determine whether we can achieve the benefits of setting learning and performance parameters from student data without exploring the full parameter space.

There are several reasons for our interest in this topic. First, as a practical matter, we are collecting data on over 50,000 students from curricula containing thousands of skills. Although there are several good algorithms for optimizing the search through the parameter space [7], finding the best fit can be computationally expensive. Different methods will typically find different parameters, and so it is important to understand whether these differences are large enough to have practical effects on the system's effectiveness.

Sensitivity to particular parameter fits may also affect generality. If the behavior of the system is relatively insensitive to the particular parameters used, then we might expect relatively little variability in these parameters as we model different cohorts of students. On the other hand, if we found extreme sensitivity, we might benefit from exploring whether different parameter sets for different groups of students might be an appropriate method to refine our modeling.

Areas of relative insensitivity within the parameter space can be used to reduce the variations in parameters that we consider. In the extreme case, if we were able to find a small number of parameter sets that provide good fits across a wide range of data, then we can exhaustively search through these parameter sets to find the best fit. Using a small number of parameter sets within the tutors, rather than searching a large space of parameters may also help us to more accurately estimate initial parameters for new units of instruction and more quickly adapt the system based on student data.

Perhaps the most interesting reason to reduce the parameter space is that it has the potential to allow us to interpret the fits that we find. The knowledge tracing algorithm can simply be thought of as a Markov process with four parameters, but we do ascribe meaning to the parameters: one represents prior knowledge, one ease of learning, one ease of guessing the answer and one the probability of slipping. When we find that the best fitting parameter set for a particular skill has a high probability of being learned, there is a tendency to believe that the data tells us that the skill is easily learned. But that interpretation could be misleading. It could be the case that the second-best fit to the data indicates a relatively low probability of being learned (with a compensating high probability of being initially known, for example). Such a case would not be a concern if there is a large difference in the quality of fit between the two parameter sets, but in an insensitive parameter space, it is quite possible that the second-best fit is almost as good a fit as the first. If that is the case, then what basis do we have for saying that the skill is easily learned (or is more difficult to learn)? If we reduce the search space, it is much easier to recognize whether there is a likely second-best fit for the skill that leads to a different interpretation. This inability to choose between parameter sets that more-or-less fit the data equally well has been called the identifiably problem [2].

Supporting interpretation of skill parameters brings us to the point where we can use parameter fitting for reasons other than optimizing knowledge tracing. For example, if we can depend on the interpretation of the *plearn* parameter, then we can identify skills that are not learned (or learned slowly) within the tutor, which gives us a metric for identifying particular skills or units of instruction that could be improved.

2 Examining the parameter space

Our data for these explorations comes from 8341 students who used at least one of four Cognitive Tutor courses (Bridge to Algebra, Algebra 1, Geometry or Algebra 2) in the 2007-08 school year. Across these four curricula, there are 2400 skills.

Our first step was to understand how the fitted parameters cover the knowledge tracing parameter space.

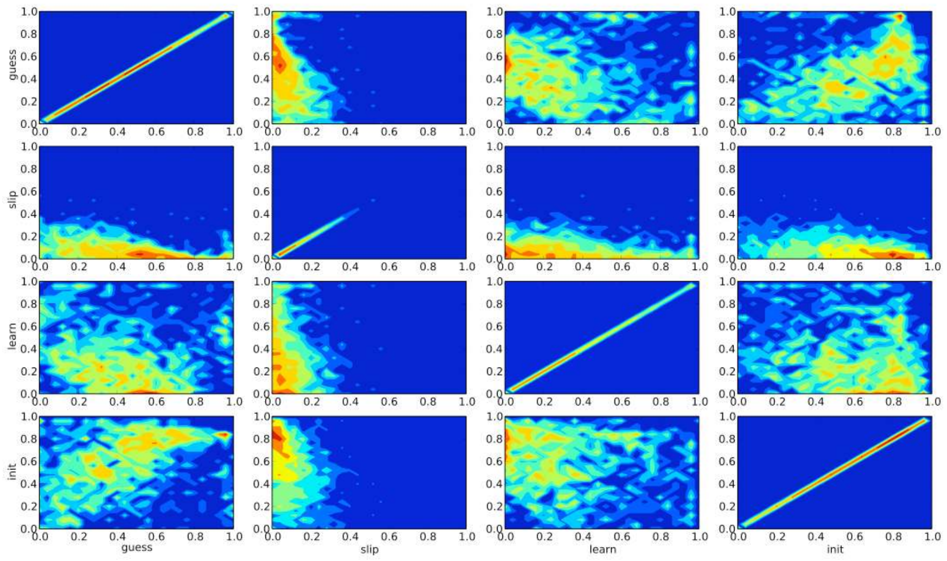

Figure 1: Heat maps showing the distribution of parameters, based on best fits of 2400 skills without constraining pguess. Each graph shows the number of skills occupying a particular position in a two-dimensional cut of the parameter space. Dark blue areas indicate regions of the space where no fits were found. Yellow and red show regions where a large number of parameter fits reside.

Figure 1 shows the results of fitting parameters on all 2400 skills. Each graph shows a two-dimensional space, defined by two of the knowledge tracing parameters. Parameters were found using an exhaustive search of the space, assuming two decimal places for each parameter (i.e. there are 100,000,000 possible parameter sets for each skill). The color in the graph indicates how many skills have a best-fitting parameter set in that region of the space. Dark blue spaces indicate areas with no skills. Yellow and red indicate areas with a large number of skills.

Inspection of Figure 1 gives us a good sense of the general shape of the parameter space. For example, it is evident that *pslip* tends to be fairly low for all skills. Skills that are judged likely to be known prior to using the tutor (with high *pinitial*) tend to have particularly low *pslip* values.

This result is promising for our goal of interpreting parameters. High values of *pslip* would be problematic for interpretation. If *pslip* exceeds 0.5, that means that the student has a greater than 50% chance of getting the item wrong, even if they know the answer. While this is logically possible, it would probably indicate a user interface where the student's intent and the student ability to express that intent in the interface are seriously compromised. If we can trust the interpretation of these parameters, then the low values of *pslip* that we see may be an indication that users generally are able to follow their intentions within the user interface.

Pguess, however, varies across the range. This is problematic. The meaning of *pguess* is the probability of being able to provide the correct answer, without having knowledge of the underlying skill. By this definition, it is hard to see how *pguess* could be greater than 0.5, because the interface never presents a case where the correct answer can be guessed with greater than a 0.5 probability. The easiest-to-guess cases in the software are ones where the student is given a two-alternative choice (0.5 probability), and those are very rare. There may be other methods of coming to a correct answer without knowledge, but they either assume that the skill model is very poor or that students generally have access to a source of answers other than their own knowledge. Baker et. al [1] call models with large values of *pguess* or *pslip* "degenerate" and also take .5 as the maximum reasonable value for these parameters. In practice, when parameters are set initially (prior to student use), we tend to fix the *pguess* parameter based on the type of question the student is being asked, with a default setting between 0.2 and 0.3.

Since part of our goal is to explore the semantics of knowledge tracing parameters, we decided to repeat this fit exercise, after constraining *pguess* to values less than 0.5. Figure 2 shows the resulting parameter space. Constraining pguess this way amounts to searching a space with 1,000,000 possible points (100 values for each of three parameters). Despite the reduction in the search space, the parameters cluster even more tightly after constraining *pguess* (that, is, there is more empty deep blue space).

The relationship between *pinitial* and *plearn* may be the most interesting, since those are the knowledge (as opposed to performance) parameters. In Figure 2, it is evident that the range of *plearn* values tends to increase as *pinitial* increases. At high values of *pinitial*, there are skills along the full range of *plearn*, and there are large clusters of skills at both very high and very low *plearn* values. This follows from the fact that high values of *pinitial* are associated with tasks that have very low error rates. If errors are infrequent, it is difficult to tell whether a particular skill is learned easily, since there are few observations of a student moving from an unlearned to a learned state. Thus, when *pinitial* is high, *plearn* can vary widely. Another way to think about this is to say that when *pinitial* is high, we cannot get a reliable estimate of *plearn*.

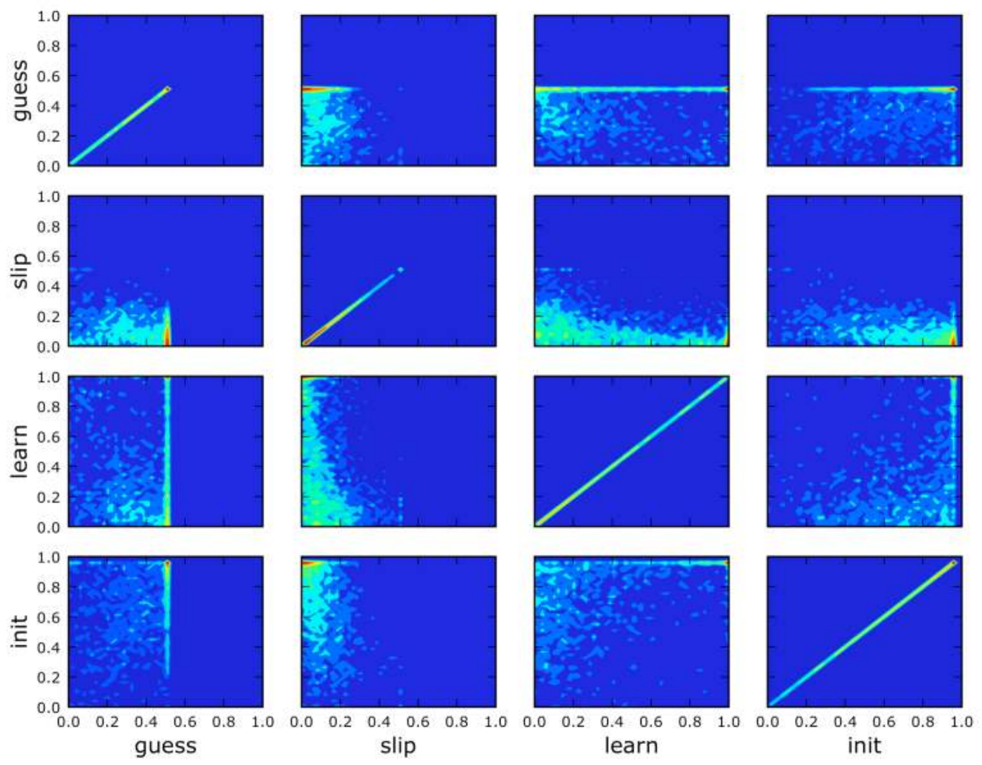

Figure 2: Parameter space with *pguess* and *pslip* constrained to be ≤ 0.5

It is also interesting to look at the relationship between *pslip* and the knowledge parameters. Although *pslip* is always low (as it was in the unconstrained fits), when *pinitial* is high, *pslip* tends to be particularly low. This finding makes sense under the assumption that skills which have been previously learned are well learned and thus relatively resistant to careless errors. Skills with both high and low *plearn*, in contrast, are in the process of being learned and thus may lend themselves more readily to slips, as is shown in the graph of the tradeoffs between those parameters.

The fact that so much of the parameter space is not used gives us hope that we will be able to find good fits to the student data using a small cluster of parameters.

3 Clustering

Building on these preliminary investigations, we set out to find the smallest group of parameter sets that could model the data sufficiently well. As a practical matter, we wanted to find a small enough number of clusters that we could imagine giving them semantically meaningful names (e.g. “not previously known but easy to learn”, “hard to learn but easy to guess”, etc.).

This approach represents a different solution to the identifiability problem than using Dirichlet priors [2]. Instead of biasing our fits based on prior beliefs about reasonable parameters, we are fitting the data using only a small number of parameter sets that provide a good fit for a large number of skills. Our assumption is that there are likely to be only a small number of semantically distinct parameter sets and that we can fit the data well using only these few sets.

In many forms of data analysis, it is assumed that a set of data was generated by some finite number of distinct processes (typically Gaussian). Clustering algorithms are a family of maximum likelihood estimation procedures for identifying these underlying processes from the set of data that they produce. The resulting model for the data consists of the set of parameters used to represent the clusters. In the current context, we are not attempting to identify the underlying generative processes (which in any event would involve complex psychological models), but rather groups of skills which behave the same with respect to the best knowledge-tracing representation. In terms of the algorithm, this turns out to mean that we are trying to identify groups of skills which project to the same regions of the p-parameter space.

In order to accomplish this, we used a k-means clustering to the fitted skills. K-means [8] is an iterative expectation-maximization [5] procedure that represents each cluster as the mean point in the parameter space. In the expectation phase, each data point is assigned to the closest cluster center. Then, in the maximization phase, each cluster center is moved to the mean point of its assigned data points. Starting with *k* cluster centers initialized at random positions throughout the parameter space, k-means converges to its final cluster positions in approximately 200-400 iterations. We used a “strict” k-means algorithm, in which the assignment of skills to clusters is an all-or-nothing relationship. This has the advantage of having a clear stopping condition – if there are no further changes in skill-to-cluster assignment, then the cluster means will not change, and the model has converged.

The K-means clustering minimizes the Euclidean distance, in the parameter space, between data points and cluster centers. This is differs from the fitting algorithm which minimizes the MSE of the predicted *pknown*, established by the model parameters, to the observed student data. Thus, it is possible to force skills into clusters that do not fit well, even though the skill is not far from the centroid of the cluster in parameter space. In theory it is possible to choose clusters that minimize the MSE to the data, rather than the distance in parameter space; in practice, however, this turns out to be computationally impractical. One avenue for future work we are looking at is ways to reduce this computational load. Since the Euclidean distance is continuous and monotonically decreasing everywhere, it is a good approximation so long as the MSE is at least locally smooth and decreasing. An informal examination of the MSE-space for a small sample of the skills indicated that this was the case, however a more in-depth examination is warranted. Using Euclidean distance has the further benefit of producing clusters that are non-disjoint in the parameter space. It would be much more difficult to justify the semantic relevance of a cluster comprised of two or more non-overlapping regions of the parameter space.

We initialized this process with $k = 50$, which converged to 23 distinct non-empty clusters. Since the assignment of skills to clusters in this particular variant of k-means is an all-or-nothing assignment, it gives the algorithm some freedom to "prune" away unnecessary clusters by assigning no data points to them. Essentially this gives the algorithm a degree of flexibility in estimating the best number of clusters needed to explain the data. Experiments with larger initial numbers of clusters (up to $k = 100$) also consistently resulted in between 20 and 25 non-empty clusters. Although the random initialization of cluster centers does introduce some variation in how the clusters converge, we found the resulting cluster centers to be very stable.

4 Interpreting the clusters

Figure 3 plots the 23 clusters that were found in the parameter space. Each cluster is represented by a circle, and the size of the circle is proportional to the number of skills that are contained in the cluster. The largest cluster contains 393 skills, and the smallest has only a single skill.

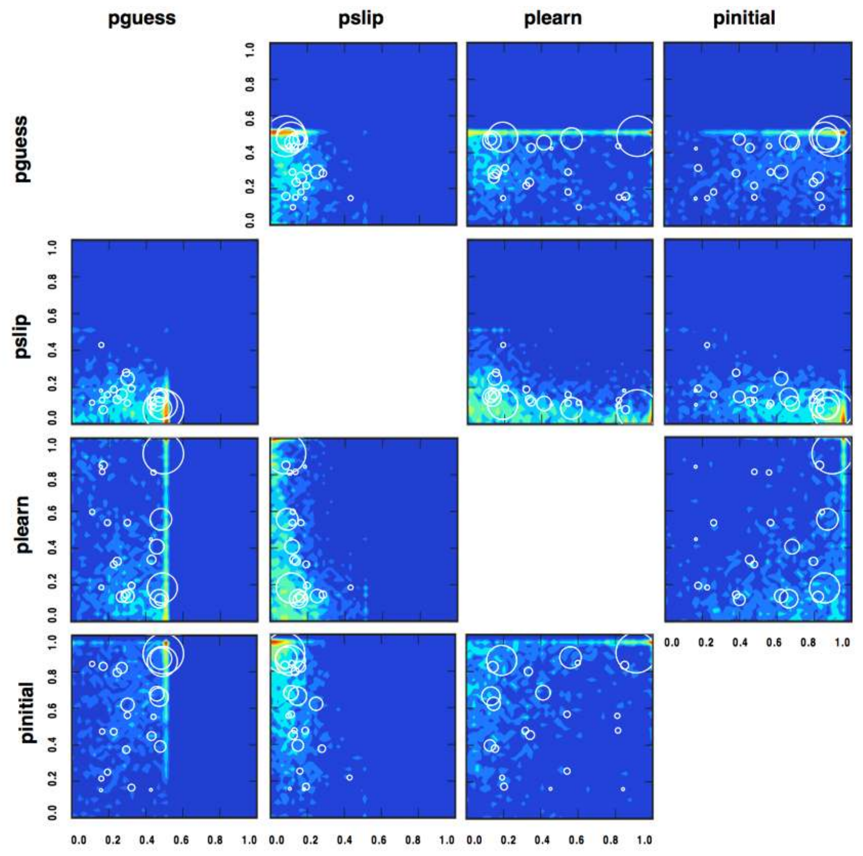

Figure 3: Positions of the final 23 clusters in the parameter space superimposed over the heatmaps. Each cluster is represented by a circle. The size of the circle is proportional to the number of skills contained in the cluster.

Using the best-fit parameters, the mean squared error (MSE) is 0.1204. Using clustered parameters increases MSE slightly, to 0.1245. MSE for the parameters delivered with the software (only some of which were set based on fits to prior years' data) was substantially higher, at 0.187. Clustering with only 23 clusters thus appears to provide a very good fit to the data, but it is difficult to understand whether even this small increase in MSE has significant effects on the behavior of the system. Since skills are bundled within problems, some skills may be presented to students even after the system has determined that the student is at mastery. For those skills, the difference between the best fit and the clustered fit may amount to nothing.

Figures 1-3 show a large number of skills with high *pinitial*. This is not surprising, since many Cognitive Tutor sections build on previous work (copying portions of a task while adding some new objectives). In these sections, skills may be repeated, and these repeats count as new skills within our model. There is little adverse effect of having skills with high *pinitial*; students will be able to master them very quickly, and their ability to master sections of the curriculum that contain a large number of skills will depend on those skills that do not have high *pinitial*. This highlights the fact that skills that are mastered quickly have little influence on system behavior. The system should be particularly insensitive to the behavior of these skills, since problem selection and mastery does not often depend on them.

For this and other reasons, Dickison et. al [6] developed a procedure for "replaying" logs of actual student behavior using fitted parameters. This algorithm takes into account skill bundling and the problem selection algorithm to determine how many problems each student in the dataset would have needed to do if the delivered parameters matched the fitted parameters. Since our goal was to predict performance in the 2008 version of the software, we used the 2008 problem selection algorithm (which changed somewhat from 2007). This necessitated dropping some sections that either incorporated changes to the skills tracked between 2007 and 2008 or that were dropped or renamed in 2008. We also excluded sections on which we had data from fewer than 10 students. For this reason, these analyses include 182 sections with a median of 177 students per section.

In order to test whether the clustered parameter sets produced substantially different system behavior than the best-fit parameter sets, we compared the median number of problems that students would need to do under best-fit parameters to the number they would need to do under the parameter sets found through clustering. The median problem counts per section using best-fit parameters were highly correlated with those using clustered parameters ($R^2 = 0.977$) suggesting that the changes in parameters made by the clustering process are negligible. The most prominent effect of clustering was that the clustered parameters often slightly reduced the change in problem count in relation to delivered parameters. The mean absolute change in median problem count (relative to the delivered parameters) was 1.95 for the fitted parameters and 1.63 for the clustered parameters. A paired t-test showed a significant difference: $t(181) = 3.2$, $p < 0.01$. This may be due to the fact that clustering tends to move parameters away from extreme values, bringing them closer to delivered parameters, which generally avoid extremes.

Another advantage of clustering is to avoid overfitting with smaller amounts of data. To test this, we developed 23 new clusters, using 1561 skills and 1312 students. We then found the best-fitting cluster for each of the 275 skills that were not used in developing the clusters, using varying numbers of students. We also found best-fitting parameters for these 275 skills on the subsets of students and tested the fit with another set of 200 students. As Figure 4 shows, when there are a small number of students contributing to the data, the clusters provide a substantially better fit to the data than the best-fit estimates. This provides evidence both that clusters developed with one set of skills will generalize to another set and that, with small amounts of student data, clusters can help prevent overfitting.

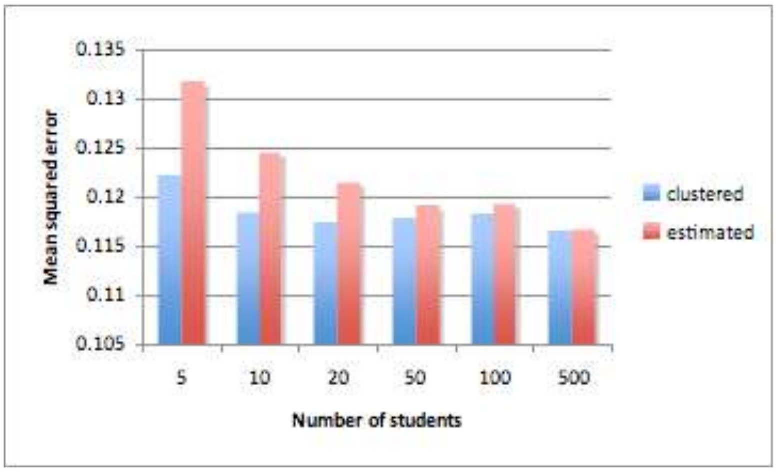

Figure 4: Comparison of clustered vs. best-fit estimates with differing numbers of students

5 Conclusion

Previous work has shown that modeling student learning and performance parameters based on prior-year student data results in improved system efficiency. This paper explored the issue of how sensitive such effectiveness is to the particular sets of parameters used. Our results have shown that tutor performance is relatively insensitive to the particular parameter sets that are used. We were able to show that, using only 23 sets of parameters, we could produce virtually the same system behavior as we would see if we had used parameters found through exploring the full parameter space. This result does not argue against fitting these parameters based on data; rather it suggests that a quick estimate of such parameters can be sufficient to produce near-optimal behavior.

It is worth pointing out that the parameters we are setting act as population parameters, which would likely benefit from adjustment for individual differences [1]. Indeed, these results may suggest that a more profitable route to accurate student modeling is to focus on individual differences, rather than population characteristics. We see clustering as complementary to both the Dirichet priors approach [2] and the use of contextual guess and slip [1].

The fact that we can model student behavior with a very small set of parameters helps us to extend the knowledge tracing model beyond simply a mathematical model of student behavior; we now have a better chance to interpret individual parameters within the set. For any knowledge component, we could calculate the goodness of fit to the data for each of the 23 parameter clusters. If we only see a good fit to one cluster, and that cluster has a high *plearn* parameter, then we can reasonably conclude that that the knowledge component is easily learned. Such a conclusion would be computationally expensive to reach in the full parameter space since, since we would need to explore a large part of the space before we could conclude that there is an almost-as-good fit to the data to be found with a low-*plearn* parameter set.

Clustering parameters thus provides us a way to quickly examine knowledge components and determine which ones are problematic. Knowledge components with low plearn might suggest areas where we should refine our instruction. Ones with high *pguess* or high *pslip* might indicate areas where we need to reconsider the user interface. Ones with high *pinitial* might indicate areas where instruction is unneeded. We are optimistic that our work in reducing the parameter space for knowledge tracing will provide us with new ways to more quickly and confidently use knowledge tracing parameters to interpret student behavior.

6 References

[1] Baker, S. J. d., Corbett, A. T. and Aleven, V. More Accurate Student Modeling Through Contextual Estimation of Slip and Guess Probabilities in Bayesian Estimation. *Proceedings of the 9th International Conference on Intelligent Tutoring Systems,* 2008, pp. 406-415.

[2] Beck, J. E. and Chang, K. M. Identifyability: A fundamental problem of student modeling. *Proceedings of the 11th International Conference on User Modeling*, 2007, pp. 137-146.

[3] Cen, H., Koedinger, K.R., Junker, B. Is Over Practice Necessary? – Improving Learning Efficiency with the Cognitive Tutor using Educational Data Mining. In Lucken, R., Koedinger, K. R. and Greer, J. (Eds). *Proceedings of the 13th International Conference on Artificial Intelligence in Education,* 2007, pp. 511-518.

[4] Corbett, A.T., Anderson, J.R. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction*, 1995, 4, 253-278.

[5] Dempster, A.P., Laird, N.M., & Rubin, D.B. Maximum Likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 1977, 39(1), 1-38.

[6] Dickison, D., Ritter, S., Harris, T. and Nixon, T. A Method for Predicting Changes in User Behavior in Cognitive Tutors. *Workshop on scalability issues in AIED 2009*.

[7] Harris, T. H., Ritter, S., Nixon, T. and Dickison, D. Hidden-Markov Modeling Methods for Skill Learning. Carnegie Learning Technical Report. 2009

[8] Lloyd, S. P., Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, 1982, 28:129-137

[9] Ritter, S., Anderson, J.R., Koedinger, K.R., & Corbett, A. The Cognitive Tutor: Applied research in mathematics education. *Psychonomics Bulletin & Review*, 2007, 14(2), pp. 249-255.