Standard Error Considerations on AFM Parameters

Guillaume Durand
National Research Council Canada
100 rue des Aboiteaux
Moncton, NB, Canada
Guillaume.Durand@nrc.ca

Cyril Goutte
National Research Council Canada
1200 Montreal Rd
Ottawa, ON, Canada
Cyril.Goutte@nrc.ca

Serge Léger
National Research Council Canada
100 rue des Aboiteaux
Moncton, NB, Canada
Serge.Leger@nrc.ca

ABSTRACT

Knowledge tracing is a fundamental area of educational data modeling that aims at gaining a better understanding of the learning occurring in tutoring systems. Knowledge tracing models fit various parameters on observed student performance and are evaluated through several goodness of fit metrics. Fitted parameter values are of crucial interest in order to diagnose learning mastery as well as knowledge models and qualitative aspects of the learning environment. Unfortunately, parameter values are rarely associated with standard errors or confidence intervals, both of which are critical information to validate the inferences that can be made from the model. Taking the example of the Additive Factor Model, we describe how to obtain standard errors on the model parameters. We propose two methods to compute those and discuss results obtained on a public dataset.

Keywords

Parameters standard error, Additive Factor Model

1. INTRODUCTION

Educational Data Mining (EDM) has already produced numerous predictive models to accurately detect, anticipate and measure meaningful outcomes of learning activities. Predicting student performance has been available for years. For instance, it was the goal of the Knowledge Discovery and Data mining (KDD) Cup 2010 [1], where teams around the world competed to get the most accurate predictions on student test item successes. While predictive accuracy and overall model goodness of fit remain central concerns, others considerations have since emerged in the EDM scientific community. Model usefulness is one of them. A model can be accurate in its predictions but useless to provide additional educational values in a learning environment [10]. Another concern, of even greater interest for the work presented in this paper, is the identifiability of the models produced and used by the EDM community. The cognitive models we use for knowledge tracing are validated towards their predictive quality but their prediction performance is not necessarily where they are most useful. This is the case, for instance, for the Additive Factor Model (AFM) [3] or the Bayesian Knowledge Tracing model (BKT) [5]. Both are widely used in intelligent tutoring systems to detect when a student has mastered a skill [15] in order to provide her with the next adequate learning material. In this situation, BKT is not used only to evaluate the probability that the student will give a correct answer at time t. It is also used to check whether the "p_known" value calculated on fitted model parameters has reached the 0.95 threshold [15]. In that case, inferring learning mastery based on fitted parameter values is risky when there is uncertainty on the fitted values. First, there is a risk that different combinations of parameters may yield functionally identical models that explain observations in the same way. This is known as the *identifiability issue*, an important problem that keeps being discussed and solved in the BKT community [2, 7]. A second issue involves the reliability and confidence in the fitted parameter values. In other words, how sure we are of the fitted parameter value that will be used to infer that the learning mastery threshold has been reached. That issue has been of primary importance in recent usage of AFM to perform advanced learning factor analysis in the field [8] or when building tools to tentatively offer guidance for building competency frameworks [9]. For instance, Durand et al. [9] describe a situation where a skill was first fitted as fairly difficult (low β) with fast learning rate (high γ). After a small modification of the training dataset, the same skill was estimated easy (large β) with no learning (small γ). In addition, it is also known from the literature that latent variable models, including skill-based cognitive models such as AFM, are difficult to estimate precisely [18]. In light of these results, it becomes crucial to take a closer look at the uncertainty on model parameters, beyond predictive accuracy. Quantifying the uncertainty on fitted parameter values by estimating their standard error appears necessary in order to increase our ability to make correct, and hopefully useful, inference from fitted models.

The rest of the article is organized as follows. The next section presents related works. Section 3 presents the AFM model, its use for diagnosing learning, and the computation of the standard error on fitted parameter values, using two different techniques. Experimental results on several cognitive models from the PSLC-Datashop [11] are presented in Section 4 and discussed in Section 5. We then summarize the contributions presented in this paper and their impact on future developments.

2. RELATED WORK

A recent and fundamental paper by Philipp et al. [17] investigates the estimation of Standard Errors in cognitive diagnostics models. Clearly identifying the need of assessing the uncertainty of the estimated model parameters using confidence intervals, they presented the theoretical background for estimating parameter standard errors for the G-DINA cognitive diagnostic model [17]. In their explanations, they essentially presented and discussed different ways of computing standard errors by either considering the complete or the incomplete information matrix. In their experiments, they managed to highlight the necessity of considering the complete information matrix rather than using the incomplete one to compute parameters standard error. This result, while interesting, was not the only focus of our interest. The authors detailed two ways of computing both the complete and incomplete information matrix in the context of G-DINA that were of primary relevance for an application to AFM. The first way uses an Outer Product of Gradient (OPG) estimator. This estimator has the advantage to be relatively easy to implement but slightly less precise than the method using the Hessian of the log-likelihood, which has the drawback of being more cumbersome to implement. In our experiments we used the Hessian estimator of the information matrix.

Computation of the standard error of parameter estimates is a classic approach in statistics method and a dense literature details its applications. However, it seems to have drawn a limited interest in the EDM community so far, as we did not find implementation examples in the EDM literature. Nevertheless, a connecting point could be found in the renewed interest on model identifiability issues [2, 7]. Identifiability issues can lead to an information matrix that is ill conditioned and that cannot be inverted. As we will see later, parameter standard error is obtained by inverting the information matrix using OPG or Hessian approaches. If the information matrix cannot be inverted, there is no standard error that can be obtained by these methods. Philip et al. mentioned that such situation can occur in the DINA model [6] whenever a "test does not involve a single-attribute item for each of the K attributes" [17]. This is a result we intuitively implemented in rules when guiding competency framework refinement with AFM [9]. Howeverm this intuitive ruleturns out to be a requirement for standard error estimation. While BKT identifiability conditions are starting to be well documented, we have not been able to find an equivalent for AFM and we hope that the scientific community will address this issue. The main objective of this contribution is to present, illustrate, and discuss the implementation of AFM parameter standard error estimation. To the best of our knowledge, this had not been addressed yet in the literature.

3. THE ADDITIVE FACTOR MODEL

The AFM [3] models the probability that a student i succeeds on an item j by a mixed-effect logistic regression:

$$P(Y_{ij} = 1|\alpha_i, \beta, \gamma) = \text{logit}^{-1}(\alpha_i + \sum_{k=1}^{K} \beta_k q_{jk} + \sum_{k=1}^{K} \gamma_k q_{jk} t_{ik}) \quad (1)$$

where $\text{logit}^{-1}(x) = 1/(1 + e^{-x})$. Parameters α_i, β_k and γ_k represent the proficiency of student i, easiness of skill k and learning rate for skill k, respectively.[1] The Q-matrix $Q = [q_{jk}]$, also known as the Knowledge Component model in the PSLC-Datashop [11], represents the item-to-skills mapping by a binary matrix, as in the following example:

$$\text{Q} = \begin{array}{c} \\ ItemA \\ ItemB \\ ItemC \\ ItemD \end{array} \begin{array}{c} \begin{array}{ccc} Skill.1 & Skill.2 & Skill.3 \end{array} \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \end{array},$$

where items A, B and D evaluate one skill each, and item C evaluates two.

Finally, variable t_{ik} is the number of times student i has practiced skill k, also known as the *opportunity* number. Parameters β and γ are key differentiators for AFM as a cognitive diagnostics model [8]. They model the learning process for each skill, making AFM a powerful and very unique model to finely characterize the acquisition of skills [8]. Learning parameters allow to plot useful *learning curves* detailing learning acquisition.

3.1 Learning curves

Learning curves are an essential tool to improve learning systems. They "give us a measure of the amount of learning that is taking place relative to the system's model" allowing to compare and improve them [14]. Concretely, a learning curve is a "graph that plots performance on a task versus the number of opportunities to practice" [14]. The performance measured can be the time spent assembling an engine component in a production line or as it is often the case in the educational field, the error rate at applying a set of, or individual skills.

Displaying learning curves in multidimensional learning environments can be difficult. Those environments are not necessary built for single skills learning measurement and they usually combine different set of skills evaluated together (multidimensionality). In such situation, we need to "retrofit" the analysis and AFM is the perfect model to do that as it tries to detect each skill specific (additive) contribution towards each item success.

Learning curves when modeling learning performance over time follow a "power law of practice" [16] which states performance over time should increase following a power law. In the Intelligent Tutoring Systems (ITSs) context, we can expect the error rate to drop as a power law over practice opportunities. Comparing ITS or sections of them can be done by considering the steepness of the curve. A steeper curve indicates a faster acquisitions of the skills practiced [14].

Another advantage of using AFM to draw learning curves is that we can compensate for the *attrition bias*. Over time, fewer learners tend to perform the items because many of them have learned the skill and the curves tend to quickly degenerate, impacting the value of slopes and the power law

[1] We refer to β and γ as the *skill* and *learning* parameters in the rest of the article.

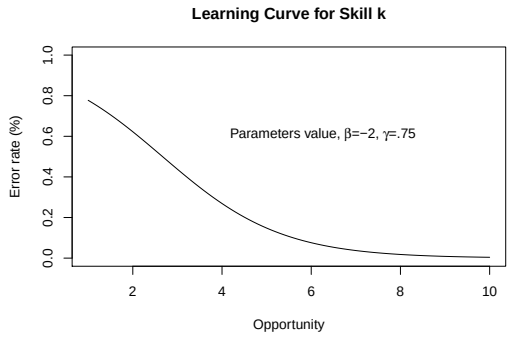

Figure 1: Example of a error curve for a moderately hard skill with a moderately fast learning rate.

fit. A convenient way to produce a learning curve for skill k in AFM is to use Eq. 1 with β_k, γ_k, and a "typical" value of the student proficiency. Using $\alpha_i = 0$ is convenient, and usually roughly corresponds to the average value of the estimated α's. This individual theoretical learning curve for skill k is given by:

$$LC_k(t) = \text{logit}^{-1}(\beta_k + \gamma_k t) = \frac{1}{1 + \exp(-\beta_k - \gamma_k t)}. \quad (2)$$

Typically, we consider *error curves* while talking about learning curves. The error curve is obtained by plotting $EC_k(t) = 1 - LC_k(t)$ as illustrated in Figure 1.

3.2 Computing the Standard Error

We present two methods to estimate the standard errors on parameters. The first one is a classical approach in the statistics literature. It involves the computation of the negative Hessian of the log-likelihood. The second one is inspired by the parametric bootstrap and estimates the standard error by computing empirical standard deviations on the parameters obtained from simulated observation samples.

3.2.1 Negative Hessian of the log-likelihood

Technically, the standard errors of estimated parameters can be retrieved from the covariance matrix of the parameters (eq. 3). More precisely, they are equal to the square root of the diagonal elements in:

$$Cov(\alpha, \beta, \gamma) = \begin{pmatrix} V_\alpha & V_{\alpha,\beta} & V_{\alpha,\gamma} \\ V_{\beta,\alpha} & V_\beta & V_{\beta,\gamma} \\ V_{\gamma,\alpha} & V_{\gamma,\beta} & V_\gamma \end{pmatrix}. \quad (3)$$

However, this covariance matrix is not known and we need to estimate it in order to compute our standard errors. Fortunately, the estimation of covariance matrices have been of interests of statisticians for a long time and several ways have been proposed to solve it. More precisely, it turns out that the covariance matrix is equal to the inverse of the information matrix [17], $Cov(\alpha, \beta, \gamma) = \mathcal{I}(\alpha, \beta, \gamma)^{-1}$. This means we can compute estimators of standard deviation on parameter estimates as long as we can compute and invert the information matrix. At the maximum likelihood, $\mathcal{I}$ is given by the negative Hessian matrix of the log-likelihood:

$$\mathcal{H}_(\mathcal{L}) = \begin{pmatrix} \frac{\partial^2 \mathcal{L}}{\partial^2 \alpha} & \frac{\partial^2 \mathcal{L}}{\partial \alpha \partial \beta} & \frac{\partial^2 \mathcal{L}}{\partial \alpha \partial \gamma} \\ \frac{\partial^2 \mathcal{L}}{\partial \beta \partial \alpha} & \frac{\partial^2 \mathcal{L}}{\partial^2 \beta} & \frac{\partial^2 \mathcal{L}}{\partial \beta \partial \gamma} \\ \frac{\partial^2 \mathcal{L}}{\partial \gamma \partial \alpha} & \frac{\partial^2 \mathcal{L}}{\partial \gamma \partial \beta} & \frac{\partial^2 \mathcal{L}}{\partial^2 \gamma} \end{pmatrix} \quad (4)$$

In our implementation of AFM, we use a penalized version of the log-likelihood, as detailed in [8], and adapt Eq. 4 accordingly.

3.2.2 Simulation

Keeping in mind that "a standard error is the standard deviation of the distribution of parameter estimates over multiple samples" [20], we simulate multiple samples from the initial data, estimate parameters on each samples, and calculate the empirical standard deviation on these results:

Algorithm 1: Pseudo-code of the simulated standard error estimation function. Values in square brackets are defaults.

Data: Q-matrix Q, first attempt observations O and α, β, γ parameter values
Parameters: Penalization parameter λ [1], number of simulations n [1000]
Result: std(α), std(β), std(γ)
Compute $P(Y_{ij} = 1|\alpha_i, \beta, \gamma)$ according to Eq. 1 for each first attempt observation O_{ij};
repeat
  Create R, a matrix of P size with random values between 0 and 1;
  Create O' a matrix equal to O;
  for *first attempt observation* O_{ij} **do**
    if $R_{ij} > P(Y_{ij})$ **then**
      $O'_{ij} \longleftarrow 0$;
  Estimate α, β, γ for each simulation iteration with respect to Q and O';
until n *simulation iterations*;
std(α) $\longleftarrow$ Standard deviation of n simulation estimated α;
std(β) $\longleftarrow$ Standard deviation of n simulation estimated β;
std(γ) $\longleftarrow$ Standard deviation of n simulation estimated γ;

This simulation approach aimed at providing us with an alternative method to validate the Hessian's detailed in previous section but also to provide us with an alternative should inverting the Hessian matrix would be impossible or too cumbersome to implement outside of our experimental environment. The simulation takes as input a Q-matrix and performance observations. It fits the AFM parameters before computing a prediction for each observation. If the prediction is below a random value uniformly distributed between 0 and 1 then the observation is changed to a failure. Then we iterate again by computing new values of AFM parameters on the new observations dataset, computing the predictions and creating another observations sample. The pseudo-code of this simulation process is presented in Algorithm 1.

We also tried another estimation method using a Jackknife approach (iterative leave-one-out on students) that provided us with overly optimistic values. Standard errors were clearly underestimated in the PSLC dataset we experimented.

Table 1: Overall predictive quality of KC models as computed by PSLC-Datashop

Model Name	KCs	#Obs.	AIC	BIC	RMSE
Arith0	18	5,104	4,948	5,569	.397095
Context	12	5,104	5,030	5,573	.399431
Original	15	5,104	5,180	5,762	.407192

4. EXPERIMENTS

4.1 Dataset

In our experiments, we used the "Geometry Area (1996-97)" dataset from DataShop [11]. It contains 6778 observations of the performance of 59 students completing 139 unique items from the "area unit" of the Geometry Cognitive Tutor course (school year 1996-1997). This is a classic Datashop collection, associated with many prior publications [3, 4, 12, 13]. We selected three Knowledge components (KCs) models to run our experiments:

- hLFASearchAICWholeModel3arith0 (Arith0 henceforth);
- hLFASearchModel1-context (Context hereafter);
- Original.

They were selected for their reasonable numbers of skills and observations but also because they have distinctive goodness of fit metrics allowing to differentiate their predictive qualities. Characteristics of these KC models, as reported in Datashop are presented in Table 1. This suggests that the best predictive model would be Arith0, followed by Context and Original. The number of skills (KCs) do not seem to correlate with the goodness of fit for these models.

4.2 Method

Our implementations are done using Matlab and Octave.[2] The AFM estimation used in previous work[8, 9], was extended with the developments described above. The Hessian of the log-likelihood was computed using an off the shelf numerical method using a central difference approximation.[3] This has the advantage of requiring no calculus for computing second derivatives, but has the disadvantage of being notably slower than direct Hessian computation. The full Hessian computation takes around three hours on a regular laptop, for each of the KC models. The simulation-based estimates were obtained using a Go language implementation of AFM parameter estimation. It takes less than 15 minutes in Go to compute 1000 simulation iterations.

4.3 Results

Table 5 shows the estimated values and standard errors for learning parameters β and γ for KC models Arith0, Context and Original. At first glance, we can see that none of the parameters take large values compared to the others. This suggests that the KC models are of excellent quality. Overall inter-model differences in parameter values and standard errors are also relatively small.

[2]Octave/Matlab implementations are available on request.
[3]Octave `Optim` package, `numhessian` function.

Table 2: Mean parameter values

KC Model	Mean parameter values		
	α	β	γ
Arith0	0(.639)	.367(1.261)	.199(.269)
Context	0(.647)	.205(1.323)	.185(.327)
Original	0(.624)	.308(.877)	.147(.127)

Table 3: Mean standard Errors computed with the Hessian

KC Model	Mean standard errors		
	α	β	γ
Arith0	.366(.149)	.349(.137)	.083(.075)
Context	.364(.149)	.320(.175)	.073(.093)
Original	.361(.149)	.284(.073)	.051(.038)

Mean parameter values (across models) in Table 2 show that all models share the same (at .001 precision) mean and almost identical standard deviations of α. This suggest that changing the KC model had a limited impact on students' proficiencies. In other words, students proficiencies remain consistently estimated from one model to another. It seems unlikely that a student proficiency would drastically change from one model to another. Interestingly the mean values of γ are higher in the better models but the standard deviation also increases suggesting higher values with more variance. If we look at the mean standard errors in Table 3, we notice that it is very similar between models for α, suggesting again a limited impact of the KC models on students proficiencies. However the values obtained for learning parameters are very interesting as the mean standard errors increase with the predictive quality of the models. One would have excepted the opposite to happen as Arith0 is expected to have a better fit of the observations than Original. In addition, standard deviations on the errors are also higher for Arith0 than Original. One assumption could be that Arith0 managed to get few better curves with more bad ones and less average good ones. More investigation would be necessary to clarify this point.

5. DISCUSSION

5.1 Model goodness of fit

The dataset used in this experiment is very adapted to conduct learning factor analysis and it is advertised as a good one to showcase PSLC-Datashop features. Consequently the discrepancy obtained between goodness of fit and mean standard error may not generalize to other situations. In addition, we have little knowledge of the intention that led to the design of these KC models. Those cautionary considerations made, we still have been able to characterize a situation were an overall better model does not necessarily lead to a a more reliable KC model. This is an interesting result, for instance, if we want to automatically refine models as in learning factor analysis as it would imply to not only look at model goodness of fit but also KC model goodness of fit. Standard errors can also inform us on the problematic skills to modify as it allow us to get a better grasp on the reliability of learning parameters for each skill.

5.2 Learning detection

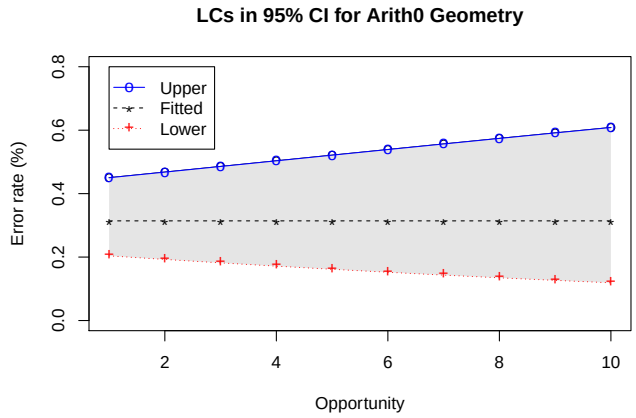

Figure 2: A skill with a flat curve suggesting limited learning for most values in the 95% confidence interval

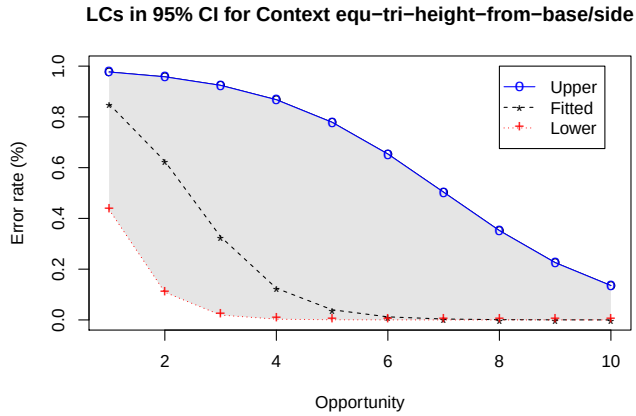

Figure 3: A skill with a steep curve clearly showing learning for all values in its 95% confidence interval

Standard errors allow us to compute confidence intervals on parameters and learning curves. Figures 2 and 3 plot learning curves for a skill with low difficulty and no learning (Fig. 2) and a difficult skill with fast learning rate (Fig. 3). In both cases, the "Fitted" learning curve uses fitted learning parameters, the "Upper" curve is obtained using the parameters at the lower end of the confidence interval ($1.96 \times$ StdErr below fitted values), and the "Lower" curve uses parameters are the top of the C.I. ($1.96 \times$ StdErr above fitted). The Upper and Lower curves provide us with the extreme slopes that the learning curve can take in a 95% confidence interval, and show the range of difficulty the skill can take while still remaining in the confidence interval.

Some values taken by these curves are not possible in practice. For instance in Figure 2, the Upper curve is impossible under AFM parameter fitting constraints, as γ is constrained to be positive. On the other hand, the Lower curve can be observed and shows limited learning. In this configuration of learning parameters, stating no learning after looking only at the Fitted learning curve could be an overstatement even though it is very likely that no learning is occurring. However as Murray et al. [15] showed, flat aggregated curves showing no learning could, in fact, hide the learning occurring for sub-group of students. In their study of an algebra curriculum containing performance data of 15,414 students on 881 skills, they discovered that around 16% of skills were misidentified as showing no learning. Standard error computation gives another reason why we should be cautious when claiming no learning. But can standard errors help us claim learning? The skill in Figure 3 answers this question. We can see that all the difficulties and slopes that can be taken in the 95% confidence interval leads to conclude that this skill is learned. In conclusion to this subsection, considering fitted parameter standard errors is important to confirm that learning is occurring but not necessarily the opposite.

Table 4: RMSE and r^2 computed between the Hessian and the simulation standard errors

	RMSE			r^2		
KC Model	α	β	γ	α	β	γ
Arith0	.052	.050	.022	.906	.963	.987
Context	.053	.061	.020	.890	.900	.973
Original	.047	.026	.004	.917	.947	.992

5.3 Simulation and Hessian methods

Table 5 shows that standard errors computed from the log-likelihood Hessian and by simulation are very close. This means that our method can potentially provide an estimate of the standard errors when the Hessian is hard to compute or invert. This also confirms the validity of our simulation results. Table 4 shows the Root Mean Square Error (RMSE) and correlation (r^2) between simulation estimates and the standard errors over all parameters of each KC Model. Although not insignificant, the difference between the two methods is sufficiently small, and the value of r^2 large enough, to consider that simulation results provide good estimates of the standard errors on parameters.

6. CONCLUSION AND FUTURE WORK

Estimating the reliability of parameter estimates is a crucial aspect of model inference. We showed how to compute standard errors on AFM model parameters, and applied the proposed methods to public datasets from the PSLC Datashop. This yields several observations.

First, the more accurate model is not always the one with the better KC model: parameter validity and predictive ability are different. That confusion is not new however and allowed progress in cognitive psychology in the first half of the nineteenth century before the community realized it failed to "provide a strong foundation for deducing likely relationships among variables, and hence for the development of generative theory"[19].

Second, standard errors, and the associated confidence intervals, provide precious insight into learning. However, characterizing the absence of learning is more complicated, especially when γ is less reliable.

Finally, standard errors on parameters can be easily estimated by the simulation method we describe. This can be

Table 5: Estimated parameters and standard errors for several PSLC models.

Model	Skill	β	StErrβ	Simul.	γ	StErr γ	Simul.
Arith0	Geometry*parallelogram-area	1.939	0.233	0.224	0.028	0.016	0.016
Arith0	Geometry*parallelogram-area*Textbk_New_Decomp...	2.540	0.617	0.659	0.180	0.149	0.192
Arith0	Geometry*Textbk_New_Decompose-circle-area	1.136	0.374	0.399	0.183	0.093	0.111
Arith0	arithmetic	1.992	0.272	0.250	0.027	0.023	0.022
Arith0	Geometry	0.781	0.260	0.197	0.000	0.036	0.021
Arith0	Geometry*decomp-trap*trapezoid-area	-0.624	0.200	0.202	0.092	0.017	0.017
Arith0	Geometry*ALT:TRIANGLE-AREA	1.501	0.341	0.260	0.000	0.056	0.035
Arith0	Geometry*ALT:TRIANGLE-AREA-PART	0.204	0.400	0.416	0.230	0.124	0.132
Arith0	Geometry*compose-by-multiplication	-0.675	0.390	0.400	0.267	0.121	0.126
Arith0	Geometry*pentagon-area	-0.550	0.199	0.200	0.110	0.015	0.016
Arith0	Geometry*ALT:CIRCLE-AREA-INDIRECT	-0.268	0.305	0.306	0.312	0.066	0.071
Arith0	Geometry*Textbk_New_Decompose-circle-area*circle...	0.871	0.255	0.258	0.073	0.030	0.031
Arith0	Geometry*ALT:CIRCLE-AREA	0.973	0.280	0.281	0.124	0.039	0.042
Arith0	Geometry*circle-area	-0.393	0.348	0.342	0.171	0.089	0.093
Arith0	Geometry*circle-diam-from-subgoal	0.126	0.275	0.268	0.071	0.045	0.043
Arith0	Geometry*equi-tri-height?	-2.986	0.714	0.888	1.232	0.310	0.385
Arith0	Geometry*decomp-trap	-0.555	0.304	0.304	0.146	0.057	0.060
Arith0	compose-subtract	0.588	0.524	0.540	0.329	0.200	0.222
Context	parallelogram-area	2.105	0.234	0.227	0.019	0.012	0.012
Context	context	0.105	0.168	0.117	0.000	0.005	0.002
Context	Geometry	0.873	0.168	0.171	0.016	0.005	0.006
Context	Subtract-rectangles	2.475	0.571	0.398	0.000	0.137	0.091
Context	decomp-trap	-0.529	0.181	0.184	0.060	0.012	0.012
Context	compose-by-multiplication	0.284	0.248	0.245	0.114	0.023	0.023
Context	pentagon-area	-0.552	0.199	0.197	0.110	0.015	0.016
Context	circle-area	0.393	0.212	0.217	0.106	0.019	0.020
Context	radius-from-area	-0.427	0.351	0.347	0.165	0.089	0.091
Context	radius-from-circumference	0.134	0.275	0.269	0.067	0.045	0.044
Context	equ-tri-height-from-base/side	-2.972	0.713	0.819	1.230	0.310	0.354
Context	Subtract	0.576	0.523	0.554	0.336	0.200	0.227
Original	ALT:PARALLELOGRAM-AREA	2.326	0.250	0.197	0.011	0.016	0.013
Original	ALT:PARALLELOGRAM-SIDE	1.054	0.494	0.473	0.345	0.152	0.157
Original	ALT:COMPOSE-BY-ADDITION	1.035	0.191	0.135	0.000	0.012	0.008
Original	ALT:TRAPEZOID-AREA	-0.860	0.344	0.340	0.344	0.092	0.094
Original	ALT:TRAPEZOID-HEIGHT	-0.800	0.329	0.340	0.243	0.079	0.083
Original	ALT:TRAPEZOID-BASE	-0.498	0.334	0.334	0.233	0.084	0.085
Original	ALT:TRIANGLE-AREA	0.964	0.249	0.237	0.042	0.028	0.027
Original	ALT:TRIANGLE-SIDE	0.122	0.297	0.245	0.037	0.056	0.044
Original	ALT:COMPOSE-BY-MULTIPLICATION	0.393	0.231	0.221	0.113	0.022	0.023
Original	ALT:PENTAGON-AREA	-1.000	0.334	0.327	0.392	0.081	0.083
Original	ALT:PENTAGON-SIDE	-0.413	0.235	0.226	0.151	0.028	0.029
Original	ALT:CIRCLE-RADIUS	0.360	0.234	0.210	0.046	0.027	0.026
Original	ALT:CIRCLE-AREA	0.473	0.209	0.197	0.104	0.019	0.020
Original	ALT:CIRCLE-CIRCUMFERENCE	0.876	0.268	0.251	0.073	0.037	0.037
Original	ALT:CIRCLE-DIAMETER	0.593	0.258	0.252	0.074	0.034	0.036

convenient when the Hessian of the log-likelihood is not easily calculated or inverted.

Our work also raised significant questions. For instance, the identifiability of the AFM model needs to be addressed, as it is likely that AFM could, like DINA be in trouble on a dataset that "does not involve a single-attribute item for each of the K attributes" [17].

7. ACKNOWLEDGMENTS

We wish to thank Mimi McLaughlin and Cindy Tipper for help and clarification with the datasets in Datashop.

8. REFERENCES

[1] KDD cup 2010, student performance evaluation. `http://www.kdd.org/kdd-cup/view/kdd-cup-2010-student-performance-evaluation/Data`[Accessed: 2018-03-07].

[2] J. E. Beck and K.-M. Chang. Identifiability: A fundamental problem of student modeling. In C. Conati, K. McCoy, and G. Paliouras, editors, *User Modeling 2007*, pages 137–146, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[3] H. Cen, K. Koedinger, and B. Junker. Learning factors analysis – A general method for cognitive model evaluation and improvement. In M. Ikeda,

K. D. Ashley, and T.-W. Chan, editors, *Intelligent Tutoring Systems: 8th International Conference, ITS 2006, Jhongli, Taiwan, June 26-30, 2006. Proceedings*, pages 164–175, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[4] H. Cen, K. Koedinger, and B. Junker. Is overpractice necessary? — Improving learning efficiency with the cognitive tutor through educational data mining. In R. Luckin, K. R. Koedinger, and J. Greer, editors, *Proceeding of the 2007 Conference on Artificial intelligence in Education: Building Technology Rich Learning Contexts that Work*, number 158 in Frontiers in Artificial Intelligence and Applications, pages 511–518, Amsterdam, Netherlands, 2007. IOS Press.

[5] A. Corbett and J. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4):253–278, 1994.

[6] J. De la Torre. DINA model and parameter estimation: A didactic. *Journal of Educational and Behavioral Statistics*, 34(1):115–130, Mar. 2009.

[7] S. Doroudi and E. Brunskill. The misidentified identifiability problem in bayesian knowledge tracing. In *Proceedings of the 10th International Conference on Educational Data Mining.*, pages 143–149. EDM, 2017.

[8] G. Durand, C. Goutte, N. Belacel, Y. Bouslimani, and S. Léger. Review, computation and application of the additive factor model (AFM). Tech. Report 23002483, National Research Council Canada, 2017.

[9] G. Durand, C. Goutte, N. Belacel, Y. Bouslimani, and S. Léger. A diagnostic tool for competency-based program engineering. In *Proceedings of the Eight International Learning Analytics & Knowledge Conference*, LAK '18, New York, NY, USA, 2018. ACM.

[10] J. Gonzalez-Brenes and Y. Huang. Your model is predictive– but is it useful? Theoretical and empirical considerations of a new paradigm for adaptive tutoring evaluation. In *The 8th International Conference on Educational Data Mining*, 2015.

[11] K. Koedinger, R. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. A data repository for the EDM community: The PSLC Datashop. In C. Romero, S. Ventura, M. Pechenizkiy, and R. Baker, editors, *Handbook of Educational Data Mining*. CRC Press, 2010.

[12] K. R. Koedinger, K. Cunningham, A. Skogsholm, and B. Leber. An open repository and analysis tools for fine-grained, longitudinal learner data. In *EDM*, pages 157–166. www.educationaldatamining.org, 2008.

[13] K. R. Koedinger, E. A. McLaughlin, and J. C. Stamper. Automated student model improvement. In *EDM*, pages 17–24. www.educationaldatamining.org, 2012.

[14] B. Martin, A. Mitrovic, K. R. Koedinger, and S. Mathan. Evaluating and improving adaptive educational systems with learning curves. *User Modeling and User-Adapted Interaction*, 21(3):249–283, Aug 2011.

[15] R. C. Murray, S. Ritter, T. Nixon, R. Schwiebert, R. G. M. Hausmann, B. Towle, S. E. Fancsali, and A. Vuong. Revealing the learning in learning curves. In H. C. Lane, K. Yacef, J. Mostow, and P. Pavlik, editors, *Artificial Intelligence in Education*, pages 473–482, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[16] A. Newell and P. S. Rosenbloom. Mechanisms of skill acquisition and the law of practice. *Cognitive skills and their acquisition*, 1:1–55, 1981.

[17] M. Philipp, C. Strobl, J. de la Torre, and A. Zeileis. On the estimation of standard errors in cognitive diagnosis models. *Journal of Educational and Behavioral Statistics*, 2017.

[18] I. Ropovik. A cautionary note on testing latent variable models. *Frontiers in Psychology*, 6:1715, 2015.

[19] M. E. Strauss and G. T. Smith. Construct validity: Advances in theory and methodology. *Annual Review of Clinical Psychology*, 5(1):1–25, 2009.

[20] T. Verguts and G. Storms. Assessing the informational value of parameter estimates in cognitive models. *Behavior Research Methods, Instruments, & Computers*, 36(1):1–10, Feb 2004.