Better Model, Worse Predictions: The Dangers in Student Model Comparisons

Jaroslav Čechák[0000−0002−8673−8705] and Radek Pelánek[0000−0001−8877−4729]

Masaryk University, Brno, Czech Republic
{xcechak1, pelanek}@fi.muni.cz

Abstract. The additive factor model is a widely used tool for analyzing educational data, yet it is often used as an off-the-shelf solution without considering implementation details. A common practice is to compare multiple additive factor models, choose the one with the best predictive accuracy, and interpret the parameters of the model as evidence of student learning. In this work, we use simulated data to show that in certain situations, this approach can lead to misleading results. Specifically, we show how student skill distribution affects estimates of other model parameters.

Keywords: Additive factor model · Student modeling · Simulation · Model comparison

1 Introduction

In order to make learning environments adaptive and personalized, we need to model the knowledge state of students [22]. Student modeling techniques are used for a variety of purposes. Models like Bayesian Knowledge Tracing or the Elo rating system are used for updating knowledge estimates after each answer, and this estimate is used for immediate personalization of the learning environment (e.g., the choice of the next question or evaluation of mastery criterion). Other types of student models are used to perform offline learning analytics, obtain actionable insights, and then use them to perform targeted interventions that improve the learning environment. This type of analysis is in literature sometimes described as "closing the loop" studies [6,13,11].

In this work, we focus on the second type of student model applications. A commonly used model for this purpose is the Additive Factor Model (AFM). The model's main aim is to evaluate and refine the domain model, specifically the mapping of items to concepts (knowledge components), which is often called Q-matrix. The term item refers to any simple task given to a student, i.e., solving a simple math problem. The concept refers to a general rule needed to correctly answer the item, i.e., the addition of natural numbers. The Q-matrix is then a binary matrix representing which items require which concepts. For each concept, AFM specifies two parameters: easiness and learning rate. Once the model is fitted to data, these parameters are interpreted as evidence of learning (or its absence). Concepts with low learning rates are natural candidates for revision.

AFM is a widely used model and has been used in a variety of previous studies. Studies [12,13,16,17,19] focus on domain model refinement, [2,15] give an overview of domain modeling using AFM, and [24] uses AFM to produce learning curves for further analysis. However, most of these studies do not pay much attention to methodological details of parameter fitting and model comparison. They often use off-the-shelf solutions like DataShop [10] without discussing implementation details and interpret the fitted model parameters as evidence of student learning. Unfortunately, in student modeling, even small methodological details can have a significant impact on the obtained results [23].

We use simulated data to explore potential problems in model comparison and interpretation of model parameters. With simulated data, we know the ground truth, and we can objectively assess the quality of fitted model parameters (which is a luxury we do not have for data coming from real students). We use our AFM implementation as well as DataShop's implementation that was used in many previous studies. We show how the treatment of student skill parameters, while rarely analyzed, can impact model comparison and values of the fitted learning rates. Specifically, we provide a concrete setting where a model with correct parameter values has worse predictive accuracy than other objectively worse models when the evaluation is done using a commonly used approach. The results show that using a black-box approach to evaluation, without proper attention to methodological details, can lead to misleading conclusions.

2 Additive Factor Model

2.1 Model Formulation

Here we formally define the additive factor model following the notation used in a recent review of the AFM [7] that is very similar across previous work. For a given group of students I and a group of items J (together with a Q-matrix mapping items to concepts), the additive factor model predicts the probability that a student i will answer an item j correctly, taking into account difficulties of concepts involved in the item j and practice history of the student i. The probability is described by the following equation:

$$P(Y_{ij}|\alpha,\beta,\gamma) = \sigma\left(z_{ij}\right) \qquad z_{ij} = \left(\alpha_i + \sum_{k=1}^{K} \beta_k q_{jk} + \sum_{k=1}^{K} \gamma_k q_{jk} t_{ik}\right) \tag{1}$$

where:

- $i \in \{1,\ldots,I\}$ is an index of a student, $j \in \{1,\ldots,J\}$ is an index of an item,
- Y_{ij} is a binary response of a student i on an item j,
- $\sigma(x) = 1/(1+e^{-x})$ is a standard logistic function,
- z_{ij} is a logit of Y_{ij},
- K is a number of concepts,

- Q is a $J \times K$ binary matrix where q_{jk} is 1 if an item j uses a skill k and 0 otherwise,
- α_i is a proficiency (prior skill) of a student i and $\overline{\alpha} \stackrel{\text{def}}{=} 0$ to avoid non-identifiability problem
- β_k is an easiness of a skill k,
- γ_k is a learning rate for a concept k, and
- t_{ik} is a number of times a student i has practiced a skill k (practice opportunities).

2.2 Parameter Estimation

In a typical application of the AFM, the Q-matrix is provided by human experts, whereas parameter vectors α, β, and γ are fitted by a parameter estimation technique. For this work, we have implemented our custom parameter estimation based on descriptions in previous work [5,7,8]. We used the TensorFlow framework to create a computational graph model for equation 1. Parameter vectors α, β, and γ are initially set to zeros and iteratively optimized using gradient descent. Initializing parameters with zeroes has the benefits of not making any ad-hoc choices, giving more reproducible results than a random initialization, and having a natural interpretation of making all probabilities of correct answers 0.5. We use a penalized log-likelihood as a cost function to optimize and Adam optimizer for computing gradients. The learning rate is gradually lowered with an exponential decay to achieve convergence more reliably. An important detail of our implementation is per concept scaling of opportunity counts into the range $[0, 1]$ as suggested in [8] to correctly fit γ values. The implementation has an option not to fit α parameters at all (in that case, they remain zero).

Previous publications often do not provide a detailed description of parameter fitting procedures used to fit AFM. It seems likely, and some explicitly mention it, that they use AFM implemented in DataShop [10]. The DataShop's AFM implementation is described in detail in [5]. The general idea can be summarized as optimization of Penalized Maximum Likelihood Estimation that penalizes high absolute α values. Our implementation is very similar in this aspect. DataShop provides two implementations in its Tigris Workflow tool: AFM[1] and Python AFM[2] Since it has been widely used, we decided to use both DataShop implementations on our simulated data. Note that DataShop is typically used to analyze real data, yet we consider this a useful test of a commonly used tool.

2.3 Treatment of Student Parameters

The primary focus of an AFM application is on getting insight into the learning domain, i.e., on the values of β and γ parameters. Even though the student parameters (prior skills α) are not necessary for the model application, they play a crucial role in proper evaluation.

[1] https://github.com/LearnSphere/WorkflowComponents/tree/dev/AnalysisAfm
[2] https://github.com/LearnSphere/WorkflowComponents/tree/dev/AnalysisPyAfm

We have reviewed some previous work on AFM to understand the typical treatment of the student parameters α. The majority of reviewed papers mention student parameters only in the formal definition of AFM. However, student parameter fitting details are often omitted, and thus their treatment in model evaluation and comparison is especially unclear.

The student parameters fitting is described in detail only in [7]; other works only hint at the general fitting method [2] or mention modeling technique for student parameters [9]. Most reviewed papers do not discuss any details of parameter fitting and probably rely on the available AFM implementation from DataShop, e.g., [12,13,19]. This claim is based on either explicit mentions of DataShop in these papers or on the visual style of learning curve figures closely matching figures produced by DataShop. Details of DataShop's AFM parameter estimation are discussed in 2.2. However, the treatment of student parameters in DataShop's model evaluation is unclear.

Commonly used evaluation metrics are Akaike information criterion (AIC) [1], Bayesian information criterion (BIC) [25], and cross-validated root mean square error (RMSE) that are used in [12,16,17,19,14,2,13]. These metrics require model predictions and, therefore, estimates of all model parameters, including the students' skills. Although it is not always reported on which dataset AIC and BIC were computed, we assumed it was done on the same data set used for training. Our experience with DataShop supports this assumption. In cross-validation, however, part of the dataset is held out during training, and it is only used later for evaluation. This poses a question, what parameters should be given to students not seen in the training data?

A straightforward solution is to use $\alpha = 0$ for unseen students, and this choice of α also makes sense for the intended use of AFM. AFM is mainly used in domain modeling (e.g., comparing Q-matrices, analyzing learning curves) and not for estimating student skills, which is typically done by other models (e.g., Bayesian Knowledge Tracing, Performance Factor Analysis). Also, estimated α parameters should be centered around zero, and so $\alpha = 0$ represents an average student. A possible alternative is to estimate α parameter after every attempt and iteratively refit the model and predict probabilities. Such an approach is, in principle, possible, but it is non-trivial, and it has not been described in previous research. For these reasons, we assume $\alpha = 0$ is used for unseen students, which is also true for our evaluations.

3 Experiments With Parameter Fitting

3.1 Data and Models

We employ simulated data in our analyses as they provide ground truth (i.e., true model parameters), otherwise not accessible for real-world datasets. The simulated datasets are generated by randomly sampling from Bernoulli distribution where the probability of a student answering an item correctly is given by equation 1. Ground truth model parameters are either hand-picked or sampled from normal distributions. We refer to the simulation setting as a scenario.

Table 1. Summary of AFM variants used in this work.

Name	Implementation	α	β	γ
AFM ground truth	custom	ground truth	ground truth	ground truth
AFM $\alpha\beta\gamma$	custom	fitted	fitted	fitted
AFM $\beta\gamma$	custom	zeros	fitted	fitted
AFM DataShop	DataShop AFM	fitted	fitted	fitted
AFM DataShop P	DataShop Python AFM	fitted	fitted	fitted

In this work, we use three similar scenarios differing in the setting of student skill distributions. They are intentionally minimalistic to highlight the effects of student skill distribution on other model parameters estimates. There is only a single concept to remove any potential effects of concept combinations on the parameter estimation. There are ten items all practicing this single concept, so the Q-matrix is only a column of ones. Student parameters α are sampled from a normal distribution centered around zero with standard deviation σ_α. The three scenarios have $\sigma_\alpha = 1$, 2, and 3, respectively. So, for example, scenario $\sigma_\alpha = 2$ has $\alpha \sim \mathcal{N}(0, 2^2)$. Parameters β and γ for the single concept are fixed to values representing a common learning situation: $\beta = -0.5$ represents a bit harder concept, and $\gamma = 0.2$ is a moderate learning rate.

In all three scenarios, we let every student attempt every item. While this is unlikely to happen in the real world, it is the best-case scenario for the model parameter estimation. With this setting, we avoid potential biases, including attrition bias [18,20] or item ordering bias [4].

We compare multiple AFMs differing in what parameters the model uses, if they were estimated, and the actual implementation of parameter estimation. All models used in this work are summarized in Table 1. Our custom implementation refers to the TensorFlow implementation discussed in 2.2. The Item average model always predicts the mean success rate observed for an item for all students.

3.2 Experimental Setup

In the experiment, we explore simulated data sets generated using scenarios described 3.1. These scenarios give us the least biased data where every student attempts every item in random order, and the main emphasis is on student parameters that we wish to explore. Using these scenarios, we have generated five training sets, each containing simulated answers of 2000 students and one testing set with simulated answers of 1000 students. Set sizes were chosen sufficiently large to reduce unwanted noise due to randomness in simulations. Note that students in the testing set do not appear in training sets. Therefore trained models have no estimate of their skills. This cross-validation setting corresponds to the real usage of student models in learning environments.

In the evaluation, we compare AFMs with ground truth parameters and AFMs fitted on training sets in terms of the predictive ability and the actual fitted parameters values. We also include the Item average model as a naïve

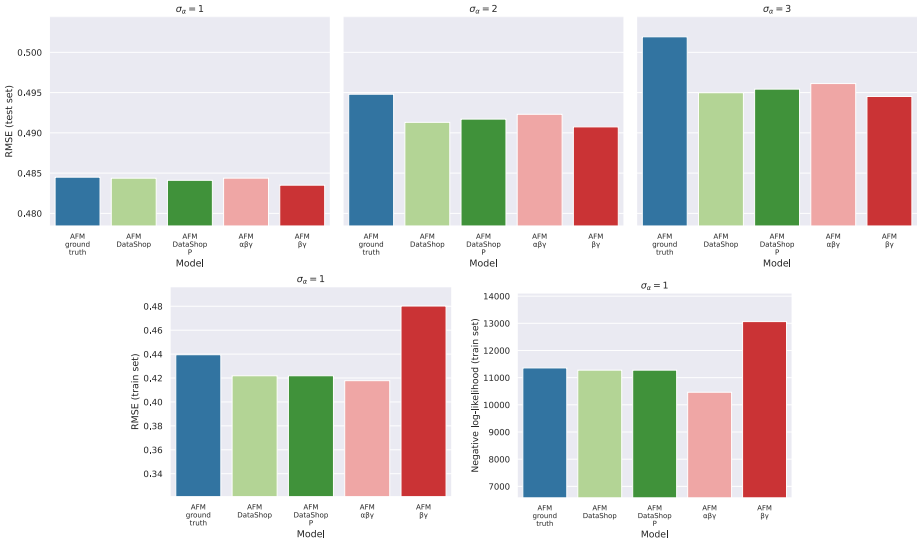

Fig. 1. Evaluation of models on training and testing sets using RMSE and negative log-likelihood. In all cases, a lower value means a better fit to data. Each bar represents the mean value of the given metric over five instances of the model fitted on five training sets. Note that y-scales do not start at zero to emphasize the relative ordering of models. Relative order based on some metric rather than the actual values is typically used in previous research to select the model that best explains the data.

baseline. All models except the two with ground truth parameters were fitted five times on five training sets and evaluated on both training and testing sets. Fitting models multiple times allows us to average out evaluation results and obtain more representative results.

For the evaluation, we choose RMSE and negative log-likelihood as our metrics. Both were used in previous research and are fairly standard [21]. RMSE is a typical choice in machine learning, and it is better suited for binary data than mean absolute error. AIC and BIC, used in previous research, are log-likelihoods with an added term for the model complexity [3]. Since we are not interested in the model complexities , we ignore the complexity term and use plain log-likelihood. RMSE is computed both on testing and training sets and log-likelihood only on training sets. When computing RMSE on the testing set, we assume all students' parameters $\alpha = 0$. The same could be done for log-likelihood, but it is typically used to measure the fit to the training data.

3.3 Results

Fig. 1 shows a comparison of models for different scenarios and different metrics. The top row of plots shows mean RMSE on the testing set, the bottom left plot shows mean RMSE on training sets, and the bottom right plot shows mean negative log-likelihood on training sets.

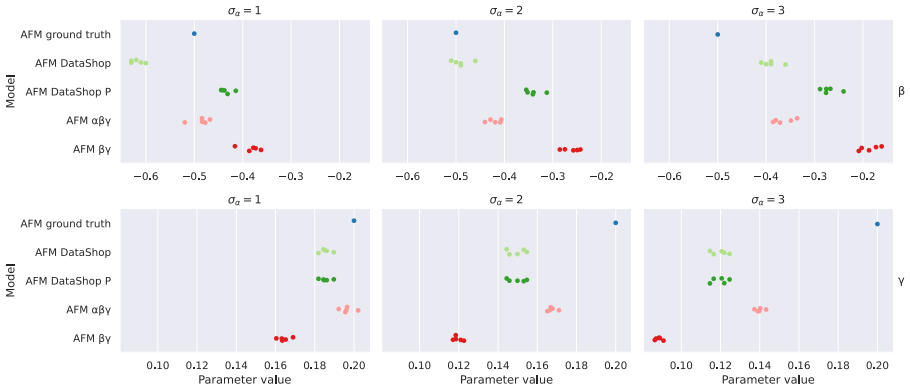

Fig. 2. Values of β and γ parameters for a single concept with increasingly wider student skill distributions. The first model uses true parameter values used in data generation. The other four models have been fitted five times on different training sets, and each dot represents an estimate from one training set. There is artificial vertical jitter to visually separate points with similar x coordinates.

Predictive performance on the testing set is comparable across most models. AFMs with fitted parameters have better RMSE than AFM ground truth in all three scenarios despite their fitted parameters differing from ground truth ones. The differences in RMSE on the testing set between AFM ground truth and models with fitted parameters are more pronounced with increasing σ_α. This suggests that models are finding a more probable explanation of data than the actual ground truth model.

AFMs with fitted student parameters also achieve much better performance on training sets both in terms of RMSE and log-likelihood. Even better than the actual model AFM ground truth that has been used to generate the data. This might suggest a slight over-fitting of student parameters.

In Fig. 2, we also examine the fitted concept parameters β and γ and compare them to ground-truth parameters. We examine three scenarios with increasingly wider student skill distributions with standard deviations $\sigma_\alpha = 1$, 2, and 3, respectively. In all three scenarios, estimated values of both β and γ are shifted towards zero. In other words, the fitted model presents the learning material as easier and with a smaller impact on learning than it is. This effect becomes stronger for wider student skill distributions.

The results also show that although there are small differences between parameters obtained by different AFM implementations, the key aspect (shift towards zero) is consistent across implementations, i.e., it is not a purely technical idiosyncrasy of a specific parameter fitting procedure. The only exception to the consistency of results is the β parameter of AFM DataShop for the scenario with $\sigma_\alpha = 1$ that is estimated further from zero. Since this model's γ parameter is estimated roughly the same as in AFM DataShop P, this could indicate a prob-

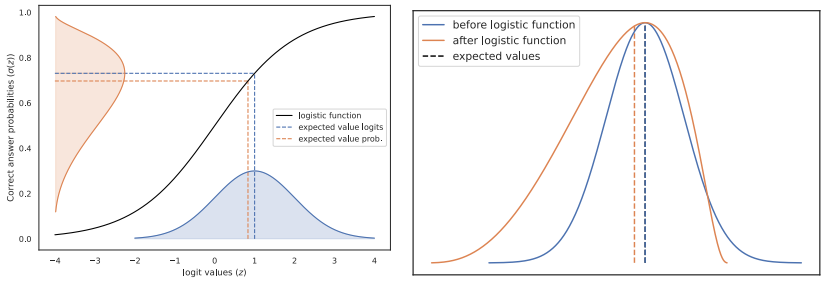

Fig. 3. An illustration of the logistic function transformation of normally distributed data and its effect on the distribution's expected value (mean). The original data follows $\mathcal{N}(1, 1)$. The left figure depicts how the distribution gets transformed, and the right figure depicts both distributions median aligned. Note that the distributions are scaled to the same height for better clarity.

lem in the implementation or data preprocessing. We were unable to pinpoint the exact source of this behavior.

3.4 Explanation

In 3.3, we observed fitted models outperforming models with ground truth parameters and estimated parameter values shifting towards zero. We believe both of these effects are caused by skewness introduced by applying the logistic function. To get probabilities of correct answers for a given item and students after a given amount of practice opportunities, normally distributed student skills are added together with concept difficulties and learning effect forming logits that are then projected using the logistic function. While logits still follow symmetric normal distribution only with shifted mean, the distribution of probabilities after applying logistic function is skewed. The median is no longer equal to the mean, as shown in Fig. 3.

The skew occurs because the logit values in one direction from the mean get projected closer together into a narrower range, and the logit values in the other direction are projected farther apart into a wider range. In Fig. 3, values to the right of the mean are squashed, and values to the left of the mean are stretched. In general, values closer to zero where the logistic function is the steepest get stretched. Values away from zero where the logistic function flattens are squashed. The skewing effect is more substantial as the mean of logits moves farther from zero, which happens for easier concepts or high opportunity counts.

When we use $\alpha = 0$ for evaluation on the testing set, we are, in effect, evaluating point estimates of correct answer probabilities (orange distribution in Fig. 3). The best point estimate with minimal RMSE is the expected value of the correct answer probability. However, due to skewness, the expected values are not the same for probabilities and logits. By estimating parameters closer to zero, models can achieve better RMSE when α is fixed to 0 and yet diverge from the true parameters simultaneously.

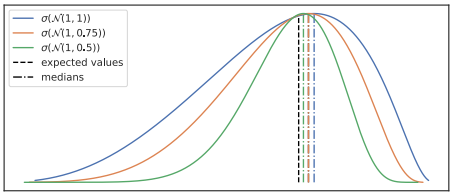

Fig. 4. An illustration of a shifting median when three normal distributions with different standard deviations get transformed by the logistic function and aligned on their expected values (means). Note that the distributions are scaled to the same height for better clarity.

The same also applies to log-likelihood if we were to evaluate it on testing set. However, we are evaluating log-likelihood on training sets with non-zero estimated α parameters, as does an optimizer during parameter fitting. After inspecting estimated values of α, we have made two observations. 1) There are only a few distinct α values, and 2) their distribution is narrower with a smaller standard deviation. The first observation is the artifact of a simple scenario with a single concept. All students with the same number of correct answers have similar estimated skills. The number of correct answers is the most differentiating factor between a student with high and low prior skill. The second observation is tied to the normalization of α parameters and explains why β and γ estimates are closer to zero. If the estimated distribution of α has a smaller standard deviation, the estimated probabilities of the correct answers are less skewed by the logistic function. After aligning expected values (means) of estimated and true probabilities of correct answers to optimize log-likelihood, we arrive at β and γ estimates closer to zero. Note that model AFM $\beta\gamma$, which keeps $\alpha = 0$, is equivalent to estimated distribution having zero standard deviation, i.e., the point estimate of expected value.

To better illustrate the effect of too strict α normalization, suppose the true probability of students answering a given item correctly after a given amount of practice follows logistic function transformed $\mathcal{N}(1, 1)$. This situation occurs when students' prior skills follow $\mathcal{N}(0, 1)$ and concept difficulties with the effect of learning sum to 1. The goal is to minimize the penalized log-likelihood. In case of too strict α normalization, the penalized log-likelihood will be optimal for α estimates with a smaller standard deviation than the true α distribution has, i.e., the standard deviation of 0.75 instead of 1, as in Fig. 4). To maximize the penalized log-likelihood, we must match the expected values of both true and estimated distributions (alignment on the black dashed line in Fig. 4). Since a narrower distribution is less skewed by the logistic function, it is necessary to shift the whole estimated distribution to make the expected values equal. The shifting of estimated distribution can only be done by changing β and γ as α parameters are constrained to be centered around zero. Thus, we have to bring β and γ parameters' estimates close to zero to maximize the penalized log-likelihood. The whole problem can be avoided by fine-tuning the penalization

hyper-parameters. However, the true student population is unknown outside of simulations, making it impossible to properly tune the hyper-parameters without cross-validation using a proper methodology on the testing set.

4 Discussion

In the previous section, we have described a specific situation where an objectively better model can lead to worse predictions than alternatives. The described situation is not a single, artificial outlier. Using simulated data, we have detected similar behavior also in other cases. For example, we simulated the ordering and mastery attrition biases, which are common in many learning systems [18,20,4], e.g., when students solve items in order from easier to more difficult and there is systematic attrition (only a subset of students solves more advanced items). Under these circumstances, even a simple baseline item average model can achieve comparable performance (with respect to predictive accuracy) as the ground truth model. Models with misspecified Q-matrix can beat the model with ground truth parameters by using the fitted parameters to compensate for the mastery bias.

These results show that fitted models have to be interpreted carefully. In our review of previous works using AFM, we have identified a common practice consisting of four steps. 1) Define several models with different candidate Q-matrices. These Q-matrices can be either constructed by experts, refined versions of Q-matrices from previous analyses, or even generated with the help of machine learning from some base Q-matrix. 2) Fit the models' parameter values on training data. Typically the training data is the same real collected data that is to be analyzed. 3) Compare the performance of the models either on the training set or the held-out testing set. To maximally utilize all available data, the evaluation is done on the training set, and k-fold cross-validation is used. 4) Select a model with the best performance and interpret its β and γ parameters (easiness and learning rate). The interpretations range from the evidence of learning to proofs of the existence of new concepts.

Our experiment with simulated data shows how this approach can lead to misleading conclusions. We showed that the best performing model is not necessarily the correct one. Models with incorrectly estimated parameters beat model with ground truth parameters due to technical issues with the treatment of the α parameters (student skills). The problem can be avoided by tuning hyper-parameters of parameter estimation procedure using cross-validation with dynamic predictions in evaluation [23], i.e., recomputing α parameters after each observed attempt in the test set (which is, however, computationally demanding).

Let us make clear that we do not claim that previously reported results are necessarily misleading. In our experiments, we purposefully use scenarios that exaggerate a specific aspect of the situation in order to clearly show its effect. Real data often contain these effects, but probably not in such a clear manner. However, our results show that more care is needed.

References

1. Akaike, H.: A new look at the statistical model identification. IEEE transactions on automatic control **19**(6), 716–723 (1974)
2. Aleven, V., Koedinger, K.R.: Knowledge component (kc) approaches to learner modeling. In: Sottilare, R.A., Graesser, A., Hu, X., Holden, H. (eds.) Design Recommendations for Intelligent Tutoring Systems, vol. 1, pp. 165–182. US Army Research Laboratory (2013)
3. Bishop, C.M.: Pattern recognition and machine learning. Springer (2006)
4. Čechák, J., Pelánek, R.: Item ordering biases in educational data. In: Proceedings of Artificial Intelligence in Education. pp. 48–58. Springer (2019)
5. Cen, H.: Generalized learning factors analysis: improving cognitive models with machine learning. Ph.D. thesis, Carnegie Mellon University (2009)
6. Cen, H., Koedinger, K.R., Junker, B.: Is over practice necessary?-improving learning efficiency with the cognitive tutor through educational data mining. In: Proceedings of Artificial Intelligence in Education. pp. 511–518 (2007)
7. Durand, G., Goutte, C., Belacel, N., Bouslimani, Y., Leger, S.: Review, computation and application of the additive factor model (afm). Tech. rep., Tech. Report 23002483. National Research Council Canada (2017)
8. Effenberger, T., Pelánek, R., Čechák, J.: Exploration of the robustness and generalizability of the additive factors model. In: Proceedings of Learning Analytics & Knowledge. pp. 472–479 (2020)
9. Käser, T., Koedinger, K., Gross, M.: Different parameters-same prediction: An analysis of learning curves. In: Proceedings of Educational Data Mining (2014)
10. Koedinger, K.R., Baker, R.S., Cunningham, K., Skogsholm, A., Leber, B., Stamper, J.: A data repository for the edm community: The pslc datashop. Handbook of educational data mining **43**, 43–56 (2010)
11. Koedinger, K.R., McLaughlin, E.A.: Closing the loop with quantitative cognitive task analysis. In: Proceedings of Educational Data Mining (2016)
12. Koedinger, K.R., McLaughlin, E.A., Stamper, J.C.: Automated student model improvement. In: Proceedings of Educational Data Mining (2012)
13. Koedinger, K.R., Stamper, J.C., McLaughlin, E.A., Nixon, T.: Using data-driven discovery of better student models to improve student learning. In: Proceedings of Artificial Intelligence in Education. pp. 421–430. Springer (2013)
14. Liu, R., Koedinger, K.R.: Variations in learning rate: Student classification based on systematic residual error patterns across practice opportunities. Proceedings of Educational Data Mining (2015)
15. Liu, R., Koedinger, K.R.: Going beyond better data prediction to create explanatory models of educational data. The Handbook of Learning Analytics pp. 69–76 (2017)
16. Liu, R., Koedinger, K.R., McLaughlin, E.A.: Interpreting model discovery and testing generalization to a new dataset. In: Proceedings of Educational Data Mining. pp. 107–113 (2014)
17. Long, Y., Holstein, K., Aleven, V.: What exactly do students learn when they practice equation solving?: refining knowledge components with the additive factors model. In: Proceedings of Learning Analytics and Knowledge. pp. 399–408. ACM (2018)
18. Murray, R.C., Ritter, S., Nixon, T., Schwiebert, R., Hausmann, R.G., Towle, B., Fancsali, S.E., Vuong, A.: Revealing the learning in learning curves. In: Proceedings of Artificial Intelligence in Education. pp. 473–482. Springer (2013)

19. Nguyen, H., Wang, Y., Stamper, J., McLaren, B.M.: Using knowledge component modeling to increase domain understanding in a digital learning game. In: Proceedings of Educational Data Mining (2019)
20. Nixon, T., Fancsali, S., Ritter, S.: The complex dynamics of aggregate learning curves. In: Proceedings of Educational Data Mining. pp. 338–339 (2013)
21. Pelánek, R.: Metrics for evaluation of student models. Journal of Educational Data Mining **7**(2), 1–19 (2015)
22. Pelánek, R.: Bayesian knowledge tracing, logistic models, and beyond: an overview of learner modeling techniques. User Modeling and User-Adapted Interaction **27**(3-5), 313–350 (2017)
23. Pelánek, R.: The details matter: methodological nuances in the evaluation of student models. User Modeling and User-Adapted Interaction **28**(3), 207–235 (2018)
24. Rivers, K., Harpstead, E., Koedinger, K.: Learning curve analysis for programming: Which concepts do students struggle with? In: Proceedings of International Computing Education Research. pp. 143–151 (2016)
25. Schwarz, G.: Estimating the dimension of a model. The Annals of Statistics **6**(2), 461–464 (1978)