Modeling Students' Learning and Variability of Performance in Problem Solving

Petr Jarušek
Masaryk University Brno
xjarusek@fi.muni.cz

Matěj Klusáček
Masaryk University Brno
matej.klusacek@mail.muni.cz

Radek Pelánek
Masaryk University Brno
pelanek@fi.muni.cz

ABSTRACT

Given data about problem solving times, how much can we automatically learn about students' and problems' characteristics? To address this question we extend a previously proposed model of problem solving times to include variability of students' performance and students' learning during sequence of problem solving tasks. We evaluate proposed models over simulated data and data from a "Problem Solving Tutor". The results show that although the models do not lead to substantially improved predictions, the learnt parameter values are meaningful and capture useful information about students and problems.

1. INTRODUCTION

In intelligent tutoring systems [1, 15] student models typically focus on correctness of student answers [6], and correspondingly the problems in tutoring systems are designed mainly with the focus on correctness. This focus is partly due to historical and technical reasons – the easiest way to collect and evaluate student responses are multiple choice questions. Thanks to the advances in technology, however, it is now relatively easy to create rich interactive problem solving activities. In such environments it is useful to analyze not only correctness of students answers, but also timing information about a solution process.

To attain a clear focus, here we consider only the information about problem solving times, i.e., we model students performance in exercises where the only performance criterium is time to solve a problem. Examples of such exercises are logic puzzles (like the well-known Sudoku puzzle) or suitably formulated programming and mathematics problems [9].

In previous work [8] we described a model which assumes a linear relationship between a problem solving skill and a logarithm of time to solve a problem, i.e., exponential relation between skill and time. In this work we present extensions of the model in two directions. The first extension models variability of performance of individual students. The second extension models students' learning (improvement of problem solving skill) during a sequence of tasks.

The feasibility of detecting students' learning and variability of performance from problem solving data depends on many factors: how much data are available, what is the noise in the problem solving times, what is the magnitude of learning effects, whether students solve problems in the same order. We use simulated data to identify conditions under which detection of students' parameters may be possible.

We also evaluate our models on real data from the Problem Solving Tutor [7, 9]. With the available data set the extended models do not bring substantial improvement of predictions of future problem solving times. Nevertheless, the experiments show that the fitted parameter values are reasonably robust and bring useful information about students' learning and variability of their performance.

2. MODELING PROBLEM SOLVING TIMES

We recapitulate the model of problem solving times which was introduced in [8] and then describe two extensions of the model and parameter estimation for these models.

2.1 Related Work

The presented models extend our previous work on modeling problem solving times [8]. The modeling approach is related to several areas. It is an example of a learner modeling in intelligent tutoring systems (see [6] for an overview) and the model is closely related to models used in item response theory [2]. Both of these areas focus mainly on modeling correctness of answers (probability that a student will answer a test item correctly), we focus on modeling timing (probability distribution of time to solve a problem). Another related area are recommender systems [10], which are used mainly in e-commerce to recommend users products that may be interesting for them. Particularly relevant technique is collaborative filtering [11], which takes matrix of ratings of products by users, and predicts future ratings. Our model is analogical to collaborative filtering when we reinterpret users as students, products as problems, and ratings as performance (for other similar work see [3, 14]). With respect to modeling learning, there is an extensive research on learning curves (e.g., [13, 12]). In the context of intelligent tutoring systems the most often used approach is Bayesian knowledge tracing [5], which models probability that a student has learned a particular skill during a sequence of attempts.

2.2 The Basic Model

We assume that we have a set of students S, a set of problems P, and data about problem solving times: t_{sp} is a logarithm of time it took student $s \in S$ to solve a problem $p \in P$. In the following we always work with a logarithm of time and use subscript s to index student parameters and subscript p to index problem parameters.

We assume that a problem solving performance depends on one latent problem solving skill θ_s and two main problem parameters: a basic difficulty of the problem b_p and a discrimination factor a_p. The basic structure of the model is simple – a linear model with Gaussian noise: $t_{sp} = b_p + a_p\theta_s + \epsilon$. Basic difficulty b describes expected solving time of a student with average skill. Discrimination factor a describes the slope of the function, i.e., it specifies how the problem distinguishes between students with different skills. Finally, ϵ is a random noise given by a normal distribution with zero mean and a constant variance (note that the model in [8] assumes problem dependent variance). The presented model is not yet identified as it suffers from the "indeterminacy of the scale" (analogically to many IRT models). This is solved by normalization – we require that the mean of all θ_s is 0 and the mean of all a_p is -1.

2.3 Modeling Variability of Students' Performance

The basic model outlined above assumes constant variance of the noise. But some students are more consistent in their performance than others and also problem characteristics influence the variance of the noise. To incorporate these factors we propose to model the variance as $c_p^2 + \sigma_s^2 a_p^2$ – a weighted sum of a problem variance c_p^2 and a student variance σ_s^2, where student's contribution to the variance depends on the discrimination of the problem. Intuitively, student's characteristics matter particularly for more discriminating problems.

Thus now we have three problem parameters a, b, c and two student parameters θ, σ. The model is the same as before, only the noise is modeled in more detail:

$$t_{sp} = b_p + a_p\theta_s + \mathcal{N}(0, c_p^2 + a_p^2\sigma_s^2)$$

The model with variance $c_p^2 + a_p^2\sigma_s^2$ is equivalent to the following approach: "at first determine a student's local skill for the attempt (based on his variance σ^2) and then determine the problem solving time with respect to this local skill":

$$p(\theta'|s) = \mathcal{N}(\theta'|\theta_s, \sigma_s^2) \qquad p(t|\theta', p) = \mathcal{N}(t|b_p + a_p\theta', c_p^2)$$

The equivalence of these two definitions is a special case of a general result for Gaussian distributions (see e.g., [4]).

2.4 Modeling Learning

It is sensible to incorporate learning into the model. The basic model assumes a fixed problem solving skill, but students problem solving skill should improve as they solve more problems – that is, after all, aim of tutoring systems. The model extension is inspired by the extensive research on learning curves (e.g., [13, 12]). A learning curve is a graph which plots the performance on a task (usually time or error rate) versus the number of trials. The shape of the learning curve is in the majority of human activities driven by power law: $T = BN^{-\alpha}$ (where T is the predicted time, N the number of trials, α the learning rate and B the performance at the first trial).

If we take the logarithm of the above mentioned form of the power law, it can be naturally combined with our basic model of problem solving times:

$$t_{sp} = b_p + a_p(\theta_s + \delta_s \cdot \log(k_{sp})) + \epsilon$$

where δ_s is a student's learning rate and k_{sp} is the order of the problem p in problem solving sequence of a student s. In the current analysis of this model we assume constant variance. Nevertheless, the model can be easily combined with the more detailed model of the noise presented above.

2.5 Parameter Estimation

We need to estimate model parameters from given data. To do so we use maximum likelihood estimation and stochastic gradient descent. As this is rather standard approach (see e.g., [4]) we focus in the following description only on the derivation of the error function and the gradient.

We derive the maximum likelihood estimation for the model with detailed noise (including student and problem variance). The likelihood of observed times t_{sp} for this model is:

$$L = \prod_{s,p} \mathcal{N}(t_{sp}|b_p + a_p\theta_s, c_p^2 + a_p^2\sigma_s^2)$$

To make the derivation more readable, we introduce the following notation: $e_{sp} = t_{sp} - (b_p + a_p\theta_s)$ (prediction error for a student s and a problem p), $v_{sp} = c_p^2 + a_p^2\sigma_s^2$ (variance for a student s and a problem p). Thus we can write the log-likelihood as:

$$\ln L = \sum_{s,p} -\frac{e_{sp}^2}{2v_{sp}} - \frac{1}{2}\ln(v_{sp}) - \frac{1}{2}\ln(2\pi)$$

Maximizing the log-likelihood is equivalent to minimizing the following error function:

$$E = \textstyle\sum_{s,p} E_{sp} \quad \text{where } E_{sp} = \frac{1}{2}\left(\frac{e_{sp}^2}{v_{sp}} + \ln(v_{sp})\right)$$

It is intractable to find the minimum analytically, but we can minimize the function using stochastic gradient descent. To do so we need to compute a gradient of E_{sp}:

$$\begin{aligned}
\frac{\partial E_{sp}}{\partial a_p} &= \frac{-e_{sp}\theta_s v_{sp} - a_p\sigma_s^2 e_{sp}^2}{v_{sp}^2} + \frac{a_p\sigma_s^2}{v_{sp}} \\
&= -\frac{e_{sp}}{v_{sp}}\left(\theta_s + a_p\sigma_s^2\frac{e_{sp}}{v_{sp}}\right) + \frac{a_p\sigma_s^2}{v_{sp}} \\
\frac{\partial E_{sp}}{\partial b_p} &= -\frac{e_{sp}}{v_{sp}} \\
\frac{\partial E_{sp}}{\partial \theta_s} &= -a_p\frac{e_{sp}}{v_{sp}} \\
\frac{\partial E_{sp}}{\partial c_p^2} &= \frac{1}{2}\left(-\frac{e_{sp}^2}{v_{sp}^2} + \frac{1}{v_{sp}}\right) = -\frac{1}{2v_{sp}^2}(e_{sp}^2 - v_{sp}) \\
\frac{\partial E_{sp}}{\partial \sigma_s^2} &= \frac{1}{2}\left(-a^2\frac{e_{sp}^2}{v_{sp}^2} + a^2\frac{1}{v_{sp}}\right) = -\frac{a^2}{2v_{sp}^2}(e_{sp}^2 - v_{sp})
\end{aligned}$$

Note that the obtained expressions have in most cases straightforward intuitive interpretation. For example the gradient with respect to θ_s is $-a_p\frac{e_{sp}}{v_{sp}}$, which means that the estimation procedure gives more weight to attempts over problems which are more discriminating and have smaller variance.

Stochastic gradient descent can find only local minima. However, by good initialization we can improve the chance of

finding a global optimum. In our case there is a straightforward way to get a good initial estimate of parameters:
b_p = mean of t_{sp} (for the given p);
a_p = 1;
θ_s = mean of $b_p - t_{sp}$ (for the given s);
$c_p = \frac{1}{2}$ of variance of $b_p - t_{sp}$ (for the given p);
$\sigma_s = \frac{1}{2}$ of variance of $b_p - t_{sp}$ (for the given s).

If we return to the original simplifying assumption and assume that the variance is constant (independent of a particular problem and student), then the error function is the basic sum-of-squares error function and the computation of gradient simplifies to $\frac{\partial E_{sp}}{\partial a_p} = -\theta_s e_{sp}$, $\frac{\partial E_{sp}}{\partial b_p} = -e_{sp}$, $\frac{\partial E_{sp}}{\partial \theta_s} = -a_p e_{sp}$. Parameter estimation for the model with learning is analogical.

3. EVALUATION

Now we report on evaluation of the models over simulated data and real data from the Problem Solving Tutor.

3.1 Simulated Data

The models described in Section 2 can be easily used to generate simulated data. Even though we have large scale data about real students, the simulated data are still useful, because for these data we know "correct answers" and thus we can thoroughly evaluate the parameter estimation procedure. The results show that the basic difficulty of problems b and students' skill θ can be estimated easily even from relatively few data. Estimating problem discrimination a is more difficult – to get a good estimate we need data about at least 30 solvers and even with more data the further improvement of the estimates is slow. As could be expected, it is most difficult to get a reasonable estimate of student and problem variance. To do so we need data about at least 50 problems and 150 students.

For the model with learning, the possibility of detection of learning depends on the situation. If the differences in learning rates are high and noise is low, then it is easy to detect the learning in the data. If the students' learning rates are very similar and noise in data is high, it is impossible to detect the learning. The feasibility of detection of learning also depends on the order in which students solve problems. In many practical cases the ordering in which students solve problems is very similar. Often students proceed from simpler problems to more difficult ones (this is certainly true for our data which are used below). If the ordering of problems for individual students is highly correlated, there is no way to distinguish between the absolute values of student learning and intrinsic difficulty of problems. On the other hand, the ordering of problems does not impact the estimation of relative learning rates (i.e., comparing students' learning rates).

3.2 Predictions

Next we report on the evaluation of predictions of problem solving times for data on real students using a Problem Solving Tutor [7, 9] – a free web-based tutoring system for practicing problem solving (available at `tutor.fi.muni.cz`). The system has more than 10 000 registered students (mainly university and high school students), who have spent more then 13 000 hours solving more than 400 000 problems. The system contains 30 types of problems, particularly computer science problems (e.g., binary numbers, robot programming, turtle graphics, introductory C and Python programming, finite automata), math problems (e.g., functions and graphs, matching expressions), and logic puzzles (e.g., Sokoban, Nurikabe, Slitherlink). For the experiment we used 8 most solved problem types from the Problem Solving Tutor, for each problem we consider only students who solved at least 15 instances of this problem.

We compare model predictions with two simpler ways to predict problem solving times. At first, we consider the mean time as a predictor – the simplest reasonable way to predict solving times (note that, consistently with the rest of the work, we compute the mean over the logarithm of time and thus the influence of outliers is limited and the mean is nearly the same as the median).

At second, we consider a simple 'personalized' predictor $\hat{t}_{sp} = m_p - \delta_s$, where m_p is the mean time for a problem p and δ_s is a "mean performance of student s with respect to other solvers", i.e., $\delta_s = (\sum m_p - t_{sp})/n_s$, where n_s is the number of problems solved by the student. Note that this corresponds to the initialization of our basic model (Section 2.5); we call it a baseline predictor.

Evaluation of model predictions was done by repeated random subsample cross-validation, with 10 repetitions. The training and testing sets are constructed in the following way: we choose randomly 30% of students and put the data of the last 20% of their attempts to the testing set; the remaining data forms the training set. Table 1. compares the results using the root mean square error metric.

The results show that the model provides improvement over the use of a mean time as a predictor. Most of the improvement in prediction is captured by the baseline model; the basic model brings a slight but consistent improvement. This improvement is larger for educational problems (e.g., Binary numbers) than for logic puzzles (e.g., Tilt maze).

Different variants (basic model with constant variance, individual variance, learning) of the model lead to similar predictions and similar values of RMSE. The model with individual variance leads in same cases to improved RMSE, the model with learning leads to slightly worse results than the basic model. One possible reason can be the higher number of parameters which causes slight overfitting. The second reason may be difference in scale between the parameters – values of the learning coefficient are typically between 0 and 0.2 while other parameters have wider spread. Thus it should be possible to improve the results of gradient descent using different step sizes for each parameter, particularly smaller step size for the learning parameter δ. Experiments with the improved algorithm really show statistically significantly better in results in some cases (e.g., in the case of Slitherlink, which is a puzzle with many opportunities for improving performance).

3.3 Analysis of Parameter Values

Even through the more complex models do not lead to substantially improved predictions, they can still bear interesting information. Predictions are useful for guiding behaviour

Table 1: Quality of predictions for different models and problems measured by root mean square error metric.

	Tilt	Robot.	Binary	Region	Slith.	Sokoban	Rush.	Nurik.
Mean time predictor	1.045	1.376	1.259	1.37	1.195	1.246	1.077	1.143
Baseline predictor	0.925	1.324	1.174	1.28	0.976	1.037	0.995	1.026
Basic model	0.92	1.301	1.148	1.28	0.948	1.021	0.981	1.025
Model with variance	0.918	1.304	1.161	1.278	0.947	1.016	0.978	1.025
Model with learning	0.948	1.313	1.181	1.322	0.967	1.034	0.993	1.04

Table 2: Spearman's correlation coefficient for parameter values obtained from two independent halves of the data.

	Tilt	Robot.	Binary	Region	Slith.	Sokoban	Rush.	Nurik.
student skill θ	0.748	0.641	0.822	0.472	0.816	0.789	0.737	0.904
student learning rate δ	0.525	0.394	0.623	0.576	0.455	0.394	0.509	0.570
basic problem difficulty b	0.994	0.961	0.951	0.927	0.981	0.963	0.962	0.837
problem discrimination a	0.469	0.564	0.569	0.282	0.533	0.347	0.434	0.195

of the tutoring systems, but small improvement in prediction precision will not change the behaviour of the system in significant way. The important aim of the more complex models is to give us additional information about students and problems, e.g., the student's learning rate, which can be used for guiding the behaviour of tutoring system and for providing feedback to users.

Since the model with learning does not improve predictions, it may be, however, that the additional parameters overfit the data and thus do not contain any valuable information. To test this hypothesis we performed the following experiment: we split the data into two disjoint halves, we use each half to train one model, and then we compare the parameter values in these two independent models. Specifically, we measure the Spearman correlation coefficient for values of each parameter.

Table 2 shows results for the model with learning. The results show, that estimates of basic difficulty and basic skill correlate highly, the weakest correlation between the estimates from the two halves is for the discrimination parameter. For students' learning rate, the additional parameter of the extended model, we get the correlation coefficient between 0.5 and 0.7 – a significant correlation which signals, that the fitted parameters contain meaningful values.

We also analyzed correlations among different model parameters, e.g., between skill θ and learning rate δ. Generally there is only weak correlation between parameters, which shows that the new parameters bring additional information.

4. REFERENCES

[1] J. Anderson, C. Boyle, and B. Reiser. Intelligent tutoring systems. *Science*, 228(4698):456–462, 1985.

[2] F. Baker. *The basics of item response theory.* University of Wisconsin, 2001.

[3] Y. Bergner, S. Droschler, G. Kortemeyer, S. Rayyan, D. Seaton, and D. Pritchard. Model-based collaborative filtering analysis of student response data: Machine-learning item response theory. In *Educational Data Mining*, pages 95–102, 2012.

[4] C. Bishop. *Pattern recognition and machine learning.* Springer, 2006.

[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] M. C. Desmarais and R. S. J. de Baker. A review of recent advances in learner and skill modeling in intelligent learning environments. *User Model. User-Adapt. Interact.*, 22(1-2):9–38, 2012.

[7] P. Jarušek and R. Pelánek. Problem response theory and its application for tutoring. In *Educational Data Mining*, pages 374–375, 2011.

[8] P. Jarušek and R. Pelánek. Analysis of a simple model of problem solving times. In *Proc. of Intelligent Tutoring Systems (ITS)*, volume 7315 of *LNCS*, pages 379–388. Springer, 2012.

[9] P. Jarušek and R. Pelánek. A web-based problem solving tool for introductory computer science. In *Proc. of Innovation and technology in computer science education*, pages 371–371. ACM, 2012.

[10] P. Kantor, F. Ricci, L. Rokach, and B. Shapira. *Recommender systems handbook.* Springer, 2010.

[11] Y. Koren and R. Bell. Advances in collaborative filtering. *Recommender Systems Handbook*, pages 145–186, 2011.

[12] 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, 2011.

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

[14] N. Thai-Nghe, T. Horváth, and L. Schmidt-Thieme. Factorization models for forecasting student performance. In *Proc. of Educational Data Mining*, pages 11–20, 2011.

[15] K. Vanlehn. The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education*, 16(3):227–265, 2006.