

Does Help Help?

A Bayes Net Approach to Modeling Tutor Interventions

Kai-min Chang, Joseph E. Beck, Jack Mostow, and Albert Corbett

Project LISTEN, School of Computer Science
Carnegie Mellon University, Pittsburgh PA, 15213, USA
kkchang@cs.cmu.edu

Abstract

This paper describes an effort to measure the effectiveness of tutor help in an intelligent tutoring system. Although conventional pre- and post- test experiments can determine whether tutor help is effective, they are expensive to conduct. Furthermore, pre- and post- test experiments often do not model student knowledge explicitly and thus are ignoring a source of information: students often request help about words they do not know. Therefore, we construct a dynamic Bayes net (which we call the Help model) that models tutor help and student knowledge in one coherent framework. The Help model distinguishes two different effects of help: *scaffolding* immediate performance vs. *teaching* persistent knowledge that improves long term performance. We train the Help model to fit student performance data gathered from usage of the Reading Tutor (Mostow & Aist, 2001). The parameters of the trained model suggest that students benefit from both the scaffolding and teaching effects of help. That is, students are more likely to perform correctly on the current attempt and learn persistent knowledge if tutor help is provided. Thus, our framework is able to distinguish two types of influence that tutor help has on the student, and can determine whether help helps learning without an explicit controlled study.

Introduction

An important property of an Intelligent Tutoring System (ITS) is its ability to help students. Thus, measuring the effectiveness of tutor help is an important evaluation of an ITS. Does help help? Does one type of help work better than the others (Heiner, Beck, & Mostow, 2004)? Even though the tentative answer is “yes” by most ITS researchers (otherwise, why include help at all in the tutor?), answering such questions is surprisingly difficult. One of the difficulties is that the question “does help help?” is ill-defined; what does it mean to help students? Does it mean to assist students in performing correctly on the current attempt or does it mean to assist in learning of persistent knowledge that will help on future attempts?

This paper describes an effort to specify how help impacts students and measure the effectiveness of tutor help in an ITS.

To measure the effectiveness of tutor help, we would ideally set up a controlled pre- and post- test experiment. A typical experimental setup works as follows: in the pre-test, we assess student performance before using the ITS. Then, we randomly assign students into two groups. The experimental group uses one version of ITS *with* the tutor help that we’re evaluating, whereas the control group uses another version of ITS *without* the particular tutor help. After students use the ITS for some time, we assess student performance of the two groups again in the post-test. We test the hypothesis that the performance improvement in the experimental group is significantly different than in the control group. This experimental design is sound and has been extensively practiced in the field of psychology. Nonetheless, the experimental design is often impractical for evaluating an ITS because a controlled experiment takes a long time to conduct and is often too expensive to conduct, although exceptions exist (e.g. Arroyo, Beck, Beal, Wing, & Woolf, 2001).

Given that the ideal pre- and post-test experimental studies are often impractical, there are several other approaches to measure the effectiveness of tutor help. For example, we may conduct user case studies and directly ask the students whether they find the tutor help effective. Unfortunately, while user case studies provide valuable qualitative feedback, they lack the ability to draw conclusive causal relationships. Alternatively, we can try to infer tutor help efficacy *from data*. For instance, one might claim that tutor help is effective if student performance improves when they receive help, compared to when they do not receive help. However, this approach raises the question of *when* to assess student performance. Immediate performance is prone to scaffolding effects where tutor help merely provides a short-term performance boost. For example, some help types provide students the answer; if students simply imitate the answer we should not count that as learning.

To measure the effect of help on persistent knowledge learning, we can use delayed performance. After all, what we care about is not just the effect of tutor help on immediate student performance, but rather its lasting effect on student knowledge. In an ITS, student knowledge is often modeled by a student model that describes the learner’s proficiencies at various skills. Indeed, tutor help and student model are two tightly coupled components in an ITS. For example, tutor help is mostly provided where the student has the least knowledge. Despite the close relationship between student model and tutor help, the two components are often evaluated and modeled independently (e.g. Heiner, Beck, & Mostow, 2004; Beck, Jia, & Mostow, 2004).

In this paper, we describe a methodology to model both tutor help and student knowledge in one coherent framework. This configuration allows us to tease apart the effect of help into 1) scaffolding immediate performance and 2) teaching persistent knowledge that improves long term performance. First, we describe prior work on assessing student knowledge using Knowledge Tracing and dynamic Bayes nets. Then, we demonstrate how dynamic Bayes nets can be used to model both tutor help and student knowledge in one coherent framework. We evaluate the proposed framework on student performance data from the Reading Tutor, an Intelligent Tutoring System that listens to children read aloud (Mostow & Aist, 2001). Finally, we conclude with contributions and future work.

Prior Work on Assessing Student Knowledge

Knowledge Tracing

Knowledge Tracing (KT) (Corbett & Anderson, 1995) is an established technique for student modeling. The goal of knowledge tracing is to estimate student knowledge from their observed actions. Prior work in this area (Beck & Sison, 2004) has used KT to model students knowledge in an ITS that uses automatic speech recognition output as its observations of student performance.

As seen in Figure 1 and Equation 1, KT maintains four parameters for each skill. Notice, KT assumes there is no forgetting, so the *forget* parameter is set to 0. Two parameters, *already know* and *learn*, are called learning parameters and refer to the student’s initial knowledge and to the probability of learning a skill given an opportunity to apply it, respectively. Two other parameters, *guess* and *slip*, are called performance parameters and account for student performance not being a perfect reflection of underlying knowledge. The *guess* parameter is the probability that a student who has not mastered the skill can generate a correct response. The *slip* parameter is used to account for even knowledgeable students making an occasional mistake.

At each successive opportunity to apply a skill, KT updates its estimated probability that the student knows the skill, based on the skill-specific learning and performance parameters and the observed student performance (evidence).

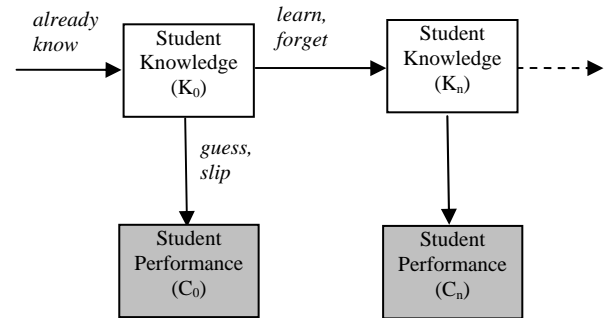


Figure 1. Graphical representation of Knowledge Tracing. Unshaded nodes represent latent variables; shaded nodes represent observed variables.

$$\begin{aligned}
 \textit{already know} &\equiv \Pr(K_0 = \textit{true}) \\
 \textit{learn} &\equiv \Pr(K_n = \textit{true} \mid K_{n-1} = \textit{false}) \\
 \textit{forget} &\equiv \Pr(K_n = \textit{false} \mid K_{n-1} = \textit{true}) = 0 \\
 \textit{guess} &\equiv \Pr(C_n = \textit{true} \mid K_n = \textit{false}) \\
 \textit{slip} &\equiv \Pr(C_n = \textit{false} \mid K_n = \textit{true})
 \end{aligned}$$

Equation 1. Parameters of Knowledge Tracing

Dynamic Bayes Net

Dynamic Bayes Net (DBN; Dean & Kanazawa, 1989) is another technique that has been applied to model student knowledge in ITS (Conati, Gertner, & VanLehn, 2002; Jonsson et al., 2005). Reye (2004) showed that KT is special case of a DBN.

In previous research, we implemented a generic Bayes net toolkit called BNT-SM (Chang, Beck, Mostow, & Corbett, 2006) for student modeling. BNT-SM inputs a data set and a compact XML specification of a DBN model hypothesized by a researcher to describe causal relationships among student knowledge and observed behavior. It generates and executes the code to train and test the model using the Bayes Net Toolbox (Murphy, 1998). BNT-SM allows researchers to easily explore different hypotheses on how is knowledge represented in a student model. We now show how to use BNT-SM to construct a DBN that models the effectiveness of tutor help on student knowledge.

Using DBNs to Assess the Effect of Help on Student Knowledge

Students often receive help on skills where they have the least knowledge. Despite the close relationship between student knowledge and tutor help, the two components are often modeled independently. In this section, we describe how to use DBNs to model both tutor help and student knowledge in one coherent framework.

Figure 2 depicts the DBN that we propose here to extend KT to include an additional *help* node, which represents whether or not the student receives tutor help prior to performing a skill. We call this network the Help model. By modeling tutor help and student knowledge in one coherent framework, the Help model allows us to tease apart two different effects of tutor help: 1) scaffolding and 2) teaching. On one hand, the effect of scaffolding is immediate in which it helps the student to perform correctly in the current attempt. On the other hand, the effect of teaching is persistent, in which it helps the student to learn and perform better on future problems. Figure 2 depicts the two effects of tutor help. Notice that Figure 2 is identical to Figure 1, with the addition of the new help node and the teach and scaffold links.

Equation 2 shows how the learning and performance parameters are computed in the Help model. Notice how the KT model and the Help model represent these parameters differently. For example, whereas KT models a single *learn* parameter, the Help model has two separate parameters conditioned on whether or not the tutor provided help: *learn/help* and *learn/no help*.

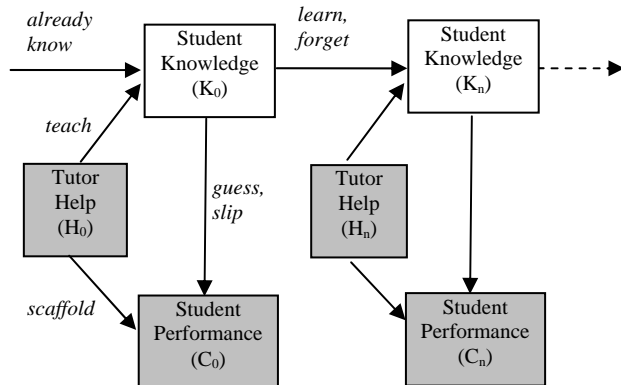


Figure 2. Graphical representation of the Help model. Unshaded nodes represent latent variables; shaded nodes represent observed variables.

$$\begin{aligned} \text{already know} | \text{help} &\equiv \Pr(K_0 = \text{true} | H_0 = \text{true}) \\ \text{already know} | \text{no help} &\equiv \Pr(K_0 = \text{true} | H_0 = \text{false}) \end{aligned}$$

$$\begin{aligned} \text{learn} | \text{help} &\equiv \Pr(K_n = \text{true} | K_{n-1} = \text{false}, H_n = \text{true}) \\ \text{learn} | \text{no help} &\equiv \Pr(K_n = \text{true} | K_{n-1} = \text{false}, H_n = \text{false}) \end{aligned}$$

$$\begin{aligned} \text{forget} | \text{help} &\equiv \Pr(K_n = \text{false} | K_{n-1} = \text{true}, H_n = \text{true}) \\ \text{forget} | \text{no help} &\equiv \Pr(K_n = \text{false} | K_{n-1} = \text{true}, H_n = \text{false}) \end{aligned}$$

$$\begin{aligned} \text{guess} | \text{help} &\equiv \Pr(C_n = \text{true} | K_n = \text{false}, H_n = \text{true}) \\ \text{guess} | \text{no help} &\equiv \Pr(C_n = \text{true} | K_n = \text{false}, H_n = \text{false}) \end{aligned}$$

$$\begin{aligned} \text{slip} | \text{help} &\equiv \Pr(C_n = \text{false} | K_n = \text{true}, H_n = \text{true}) \\ \text{slip} | \text{no help} &\equiv \Pr(C_n = \text{false} | K_n = \text{true}, H_n = \text{false}) \end{aligned}$$

Equation 2. Parameters of the Help model

Evaluating the Student Model and the Effect of Tutor Help

Given the proposed Help model, we now train the model to fit the student performance data gathered from usage of the Reading Tutor. Our evaluation of the Help model is two fold. First, we evaluate the student model by comparing the model fit of the Help model to that of the original KT model trained on the same data. Secondly, we evaluate the effectiveness of tutor help. The trained parameters for the Help model allow us to do so by seeing whether help impacts the learn parameters (the teaching effect) or the guess parameters (the scaffolding effect).

Data Collection

Our data came from 360 children between six and eight years old who used Project LISTEN's Reading Tutor (Mostow & Aist, 2001) in the 2002-2003 school year. Over the course of the school year, these students read approximately 1.95 million words (as heard by the automatic speech recognizer). On average, students used the tutor for 8.5 hours. During a session with the Reading Tutor, the tutor presented one sentence (or fragment) at a time for the student to read aloud. The student's speech was segmented into utterances delimited by silences. Each utterance was processed by the Automatic Speech Recognizer (ASR) and aligned against the sentence. This alignment scored each word of the sentence as either accepted (classified by the ASR as read correctly) or rejected (thought to be misread or omitted). ASR acceptance is modeled as the observed performance (C_n) in both the Help model and the KT model. Furthermore, the Help model receives an additional source of information – whether or not students receive help prior to the reading attempt (H_n). For modeling purposes, this paper treats each English word as a separate skill.

Parameter Estimation

We first separated the training and testing sets by splitting the students into two groups. We sorted the students according to their amount of Reading Tutor usage and then alternately assigned students to the two sets. We used the Expectation Maximization (EM) algorithm to optimize data likelihood (i.e. the probability of observing our student performance data). EM is the standard algorithm used in the machine learning community to estimate DBN parameters when the structure is known and there exist latent variables (e.g., student knowledge, K_n). EM is guaranteed to converge to a local maximum on the likelihood surface. We used the junction tree procedure for exact inference (estimating the value of the hidden variables). See Jensen (2001) for a thorough introduction to Bayes nets and the standard training and inference algorithms. In our evaluation, we trained two DBNs with the structure of the traditional KT model and the Help model.

Evaluation of Student Model

Since student knowledge is a latent variable that cannot be directly observed, we have no gold standard to compare against. Instead, we used the trained student model to predict whether the ASR would accept or reject a student's next attempt of the problem. That is, we observe reading item by item and predict whether the next word will be read correctly (in not yet seen test data). An ROC (Receiver Operating Characteristic) curve measures the performance of a binary classifier by plotting the true positive rate against the false positive rate for varying decision thresholds. The area under the ROC curve (AUC) is a reasonable performance metric for classifier systems, assuming no knowledge of the true ratio of misclassification costs (Hand, Mannila, & Smyth, 2001).

To evaluate the accuracy of our model's internal estimate of student knowledge, we compare both the AUC and the correlation between its estimates and student performance (as scored by the ASR) on the held out test data.

As Table 1 shows, the simpler KT model outperforms the Help model on both the correlation and the AUC evaluation metrics. Although the two correlation coefficients are reliably different at $p < 0.01$, their values are quite comparable. The values of model fit appear low because we are predicting individual student performance data rather than aggregated performance. It is difficult to predict a student's individual responses. We performed a cheating experiment to determine the best correlation with student performance that a student model could achieve. The cheating experiment allowed the student model to peek at future data before making a prediction. However, the cheating model was limited in that it had to make the same prediction for the current item as it made for the prior item, unless its last prediction was incorrect. Another way of thinking of this model is that it

is monotonic. That is, if it predicts a student will get an item correct and he gets it correct, the model cannot then backtrack and predict he will get the next item incorrect. It must first receive evidence that it has over- or underestimated the student's knowledge before changing its prediction. The cheating experiment revealed that the maximum correlation of any model that obeyed monotonicity constraints was only 0.5. Note that this maximum performance requires peeking at the data to be predicted and is not necessarily attainable by any actual model.

We were troubled by the observation that the Help model did not fit the student performance data better than the KT model, which ignored the tutor intervention data. One possible explanation for the relatively weaker predictive accuracy of the Help model is that it was over-fitting the data. Over-fitting occurs when a model has too many parameters relative to the number of data. As a result, the model tunes the parameter overly well to the training data and does not generalize as well to held out data. In our case, the Help model has more parameters to estimate (8 parameters per skill for roughly 3000 skills) than the KT model (4 parameters per skill). To examine whether or not the Help model was over-fitting to the training set, we reused the obtained KT and Help model to estimate student knowledge on the training data. Table 1 shows a clear over-fitting pattern. The Help model yields a better model fit (in both correlation and AUC measures) than the KT model on the *training* data, but a worse model fit on the *held out* data.

Table 1. Comparing the model fit of the KT model and the Help model on both the held out testing data and the training data

	Held out testing data		Training data	
	Correlation	AUC	Correlation	AUC
KT	0.144	0.615	0.197	0.652
Help	0.135	0.612	0.201	0.654

Evaluation of the Effectiveness of Tutor Help

To evaluate the effectiveness of tutor help, we now compare the model parameters estimated by both the KT model and the Help model.

Table 2 shows the parameters estimated for the KT model and the Help model, respectively. Notice that the KT model does not consider the help information, whereas the Help model has the parameters conditioned on whether or not tutor help is given or not. Thus, even though the Help model has worse model fit than the KT model, the Help model is more informative since it distinguishes two types of influence that help has on the student. Notice that model informativeness is sometimes at odds with model fit. For example, structural equation models usually do not fit data as well as simple regression, but are more interpretable and lend themselves to create new hypotheses

(Cohen, 1995). In this case, the greater informativeness outweighs the slight decrease in model fit.

As seen in the Help model of Table 2, the probability of *already know* is much higher when there is no help than when there is help. This suggests that tutor help is more likely to be provided for harder words – a positive finding. Also, the probability of *learning* is higher when there is help than when there is no help. Even though the difference is small, it is at least in the right direction, suggesting that tutor help does have a learning effect on long term knowledge acquisition.

Table 2. Comparing the parameters estimated by the KT model and the Help model

	KT model	Help model	
		No Help Given	Help Given
Already know	0.618	0.660	0.278
Learn	0.077	0.083	0.088
Guess	0.689	0.655	0.944
Slip	0.056	0.058	0.009

Also seen in the Help model in Table 2, the probability of guess is higher when there is help than when there is no help and the probability of slip is higher when there is no help than when there is help. This finding suggests that tutor help does have a scaffolding effect on assisting immediate performance. Notice that, although we have argued that learning effect is more beneficial in the long run than the scaffolding effect, we cannot ignore the latter. For instance, if a student is stuck when using the tutor, the tutor should still help the student to become unstuck even if that means providing the answers. Making sure that students do not get stuck is important to continue use of the tutor. Thus, it is also important that we observe the scaffolding effect.

Finally, both the learning and scaffolding effects are statistically reliable at $p < 0.05$ (paired samples t-test, weighted by number of observations for each skill), suggesting that tutor help does have an effect on both student knowledge and student performance.

Contributions

In this paper, we have proposed a methodology to infer the efficacy of help from observational data rather than experimental data. We propose a dynamic Bayes net to evaluate the effectiveness of tutor help without an explicit controlled study.

The second contribution this paper makes is on simultaneous representation of tutor help and the student model. Previous approaches addressed these problems separately by ignoring one to solve the other (Heiner, Beck, & Mostow, 2004; Beck, Jia, & Mostow,

2004). Specifically, KT ignored help, and some other experiments (e.g. Mostow & Aist, 2001) ignored student knowledge, or how it changed over time. Notice that, Jonsson et al. (2005) also used dynamic Bayes nets to model both tutor help and student knowledge simultaneously. There are, however, two differences between their approach and our approach. First, they did not distinguish the scaffolding and teaching effect in their model. Rather, they assumed that tutor help always impacts student knowledge and did not model the scaffolding link. Secondly, we started with the existing Knowledge Tracing technique and show how to extend KT to model tutor help.

The third contribution this paper makes is on distinguishing between two effects of help: scaffolding immediate performance vs. boosting persistent learning of help. Prior work either assumed help has no direct impact on student learning (Conati, Gertner, & VanLehn, 2002) or that help has no direct impact on student performance (Jonsson et al., 2005). Moreover, because we model tutor help and student knowledge in one coherent framework, we can estimate the scaffolding and learning effect. This separation of immediate vs. persistent effect of help allows researchers to understand what the tutor intervention is really doing. For instance, it is possible to investigate whether some tutor interventions help persistent learning while others mainly help immediate performance.

Future Work and Conclusions

Currently, due to limitations in BNT-SM, we could only test models with discrete, binary variables. For example, in the Help model, we only answer the question “*does help help at all?*” A more interesting question to ask is “*which type of help helps better and when.*” Thus, a future study is to extend BNT-SM to handle multi-nominal variables, which allows modeling of different help types.

One question that we are interested to explore is how does our dynamic Bayes net framework compare to the pre- and post- test experimental design (Arroyo, Beck, Beal, Wing, & Woolf, 2001). Do they draw similar conclusions, despite the fact that an experimental design is usually more expensive to conduct than data fitting with DBNs? Moreover, what kinds of causal relationship can we infer with Bayes nets?

Another issue that we are interested to address is the over-fitting pattern that we observe. Over-fitting is usually addressed by reducing the number of free parameters in the model. One solution to this problem is to construct a hierarchical model where help has the same impact on all words. That is, we would estimate one scaffolding and one teaching parameter for all skills, rather than estimating the two parameters *per skill*.

Although our study has mainly focused on modeling tutor help, our methodology can be applied to any kind of tutor intervention in general. Our work focuses on item level help (largely student initiated) because that is where we have the most data. This work is a challenging data mining problem because Bayes nets require a huge amount of data to estimate the model parameters reliably.

This paper describes an effort to measure the effectiveness of tutor help in an intelligent tutoring system. We propose a dynamic Bayes net to model tutor help and student knowledge in one coherent framework. Even though the additional information in the Help model does not yield a superior model fit, its informativeness outweighs the slight decrease in model fit. That is, the Help model distinguishes two types of influence that tutor help has on the student, and can determine whether help helps learning without an explicit controlled study.

Acknowledgements

This work was supported by the National Science Foundation, ITR/IERI Grant No. REC-0326153. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation or the official policies, either expressed or implied, of the sponsors or of the United States Government. We also acknowledge members of Project LISTEN who contributed to the design and development of the Reading Tutor, and the schools that used the tutor.

References

Arroyo, I., Beck, J. E., Beal, C. R., Wing, R. E., & Woolf, B. P. (2001). *Analyzing students' response to help provision in an elementary mathematics Intelligent Tutoring System*. Paper in Help Provision and Help Seeking in Interactive Learning Environments. Workshop at the Tenth International Conference on Artificial Intelligence in Education.

Beck, J. E., Jia, P., & Mostow, J. (2004). Automatically assessing oral reading fluency in a computer tutor that listens. *Technology, Instruction, Cognition and Learning*, 2(1-2), 61-81.

Beck, J. E., & Sison, J. (2004, September 1-3). *Using knowledge tracing to measure student reading proficiencies*. Paper in Proceedings of the 7th International Conference on Intelligent Tutoring Systems, p. 624-634

Chang, K.-m. K., Beck, J. E., Mostow, J., & Corbett, A. (2006). *A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems*. Paper in 8th International Conference on Intelligent Tutoring Systems, to appear

Cohen, P. R. (1995). *Empirical Methods for Artificial Intelligence*. Cambridge, Massachusetts: MIT Press.

Conati, C., Gertner, A., & VanLehn, K. (2002). Using Bayesian Networks to Manage Uncertainty in Student Modeling. *User Modeling and User-Adapted Interaction*, 12(4), 371-417.

Corbett, A., & Anderson, J. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4, 253-278.

Dean, T., & Kanazawa, K. (1989). A model for reasoning about persistence and causation. *International Journal of Computational Intelligence*, 5(3), 142-150.

Hand, D., Mannila, H., & Smyth, P. (2001). *Principles of Data Mining*. Cambridge, Massachusetts: MIT Press.

Heiner, C., Beck, J. E., & Mostow, J. (2004, June 17-19). *Improving the help selection policy in a Reading Tutor that listens*. Paper in Proceedings of the InSTIL/ICALL Symposium on NLP and Speech Technologies in Advanced Language Learning Systems, p. 195-198

Jensen, F. V. (Ed.). (2001). *Bayesian Networks and Decision Graphs*: Springer.

Jonsson, A., Johns, J., Mehranian, H., Arroyo, I., Woolf, B., Barto, A., Fisher, D., & Mahadevan, S. (2005, July 10). *Evaluating the Feasibility of Learning Student Models from Data*. Paper in Educational Data Mining: Papers from the AAAI Workshop, p. 1-6

Mostow, J., & Aist, G. (2001). Evaluating tutors that listen: An overview of Project LISTEN. In K. Forbus & P. Feltovich (Eds.), *Smart Machines in Education* (pp. 169-234). Menlo Park, CA: MIT/AAAI Press.

Murphy, K. (1998). *Bayes Net Toolbox for Matlab*. Retrieved March 21, 2006, from the World Wide Web: <http://bnt.sourceforge.net>

Reye, J. (2004). Student Modelling based on Belief Networks. *International Journal of Artificial Intelligence in Education*, 14, 1-33.