COGNITIVE SCIENCE
A Multidisciplinary Journal

Cognitive Science 44 (2020) e12900

ISSN: 1551-6709 online
DOI: 10.1111/cogs.12900

Assessing Mathematics Misunderstandings via Bayesian Inverse Planning

Anna N. Rafferty,[a] Rachel A. Jansen,[b] Thomas L. Griffiths[c]

[a] *Department of Computer Science, Carleton College*
[b] *Department of Psychology, University of California*
[c] *Departments of Psychology and Computer Science, Princeton University*

Received 20 October 2017; received in revised form 14 July 2020; accepted 21 July 2020

Abstract

Online educational technologies offer opportunities for providing individualized feedback and detailed profiles of students' skills. Yet many technologies for mathematics education assess students based only on the correctness of either their final answers or responses to individual steps. In contrast, examining the choices students make for how to solve the equation and the ways in which they might answer incorrectly offers the opportunity to obtain a more nuanced perspective of their algebra skills. To automatically make sense of step-by-step solutions, we propose a Bayesian inverse planning model for equation solving that computes an assessment of a learner's skills based on her pattern of errors in individual steps and her choices about what sequence of problem-solving steps to take. Bayesian inverse planning builds on existing machine learning tools to create a generative model relating (mis)-understandings to equation solving choices. Two behavioral experiments demonstrate that the model can interpret people's equation solving and that its assessments are consistent with those of experienced teachers. A third experiment uses this model to tailor guidance for learners based on individual differences in misunderstandings, closing the loop between assessing understanding, and using that assessment within an educational technology. Finally, because the bottleneck in applying inverse planning to a new domain is in creating the model of possible student misunderstandings, we show how to combine inverse planning with an existing production rule model to make inferences about student misunderstandings of fraction arithmetic.

Keywords: Inverse planning; Equation solving; Computational modeling; Markov decision processes

Correspondence should be sent to Anna N. Rafferty, Department of Computer Science, Carleton College, Northfield, MN 55057. E-mail: arafferty@carleton.edu

Parts of this work were previously presented at the 17th International Conference on Artificial Intelligence in Education (Rafferty & Griffiths, 2015) and at the 9th International Conference on Educational Data Mining (Rafferty, Jansen, & Griffiths, 2016).

1. Introduction

When teachers observe individual students working on problems, they can gain great insight into each student's particular understanding (and misunderstandings) about a topic. This insight comes from observing students' strategies and problem-solving steps. Even if two students give the same answer to a problem, the way they reached that solution may sallow the teacher to form different assessments about what each understands and does not understand. For example, a teacher might observe two students give the same wrong answer to an algebra equation, but by watching how the students solved the equation, recognize that one was struggling with how to divide fractions, whereas the other had a misunderstanding related to what types of terms can be combined. However, in many online settings, students are only evaluated based on how many problems they answer correctly. For example, websites such as Khan Academy typically allow students to move on when they have completed a particular number of problems in a row; similarly, ASSISTments is a common platform for mathematics homework where students may be assigned to complete problems until they get three right in a row (Heffernan & Heffernan, 2014).[1] Checking correctness for math problems where students give a final numeric answer is simple to do automatically, but focusing only on correctness of final answers misses the nuances that can be understood by looking more closely at students' problem-solving processes.

There are a number of automated tutoring systems for mathematics that look more closely at individual problem-solving steps. For example, Cognitive Tutor Algebra (Koedinger, Anderson, Hadley, & Mark, 1997) is one system where students enter in step-by-step data as they solve problems, and the system uses a cognitive model to make inferences about individual students, estimating whether each student has mastered each of a discrete set of skills. These systems often structure problem solving in somewhat restrictive ways, such as requiring students to get each individual step of a problem correct before continuing, to complete steps in a fixed order, or to indicate what action they are taking to go from one step to another. This can be beneficial for providing step-by-step immediate feedback and may assist in making automated inferences about students' knowledge. Outside of mathematics, some older work found that college students expressed similar satisfaction with tutor interfaces that did not flag their errors, flagged their errors but did not force them to correct those errors, or flagged their errors and forced correction (Corbett & Anderson, 1990), although in some cases, immediately flagging errors negatively impacted learning (Schooler & Anderson, 1990). However, structured problem solving could also be frustrating to some learners. Previous work comparing a typical interface to a handwriting interface for Cognitive Tutor Algebra (Koedinger et al., 1997), where the handwriting interface only provided feedback on students' final answers, found that while step-by-step feedback was more beneficial for learning (Anthony, 2008), students preferred the handwritten interface (Anthony, Yang, & Koedinger, 2005). Beyond the efficiency of handwriting, students may also have appreciated being able to show their work without constraints on step correctness. Rigid structuring of the ordering of steps or their correctness may lead some users to disengage, potentially outweighing the benefits of immediate feedback. Additionally, this structuring limits the ability to make inferences about a student's understanding based on her pattern of choices, since those choices are constrained by the system.

Providing students with the freedom to make choices when solving problems, without immediate intervention, may also be helpful for permitting learners to work through the many ideas they may have about a particular domain and that may resurface in different contexts. Within scientific domains, knowledge integration theories have been used to design curricula that elicit student ideas and help students distinguish among these ideas to build a more complete and consistent understanding (Linn, Lee, Tinker, Husic, & Chiu, 2006). While our focus is not on science domains, students have also been observed to have multiple conflicting ideas about equation solving (e.g., Sleeman, 1984). By allowing learners to demonstrate many different understandings, collecting evidence about their patterns of behavior which are linked to those understandings, an automated system can provide more general opportunities for students to sort out their beliefs, supporting an integrated understanding.

In this paper, we explore an alternative system that allows students to freely choose sequences of steps to take when solving problems. Past work has used Bayesian inverse planning (Rafferty, LaMar, & Griffiths, 2015) to model people's understanding based on sequences of steps in laboratory experiments and educational games, showing a way to make fine-grained assessments that identify misunderstandings based on patterns of actions. Our main contribution in this work is to develop a Bayesian inverse planning approach for modeling students' problem solving in mathematical domains, primarily algebraic equation solving. The Bayesian inverse planning model is related to other Bayesian models of human cognition that have been used to create generative models of people's learning and generalization in experimental psychology (e.g., Griffiths, Chater, Kemp, Perfors, & Tenenbaum, 2010; Lee, 2011; Tenenbaum, Kemp, Griffiths, & Goodman, 2011). These models can use sophisticated representations of possible ways of understanding and provide close fits to human performance in online and laboratory experiments. However, there have been limited applications of these models to behavior in more typical educational domains (e.g., Lindsey, Shroyer, Pashler, & Mozer, 2014). We primarily focus on algebraic equation solving as a place to explore the use of a Bayesian cognitive model that allows graded representation of understanding both because of the technical challenges introduced by this domain and because of the importance of algebra within mathematics education and as a gatekeeper to higher education.

From a theoretical perspective, algebraic equation solving offers sufficient complexity to explore how models that work well in the laboratory translate to an authentic educational context. Inverse planning has previously been used to model people's choices (Rafferty et al., 2015) and as a measurement model in educational assessment (LaMar, 2017), but algebraic equation solving requires a somewhat modified algorithm from past work to make inferences about understanding based on the model, as actions are not directly observed but must be inferred from what a student writes down and there are an infinite number of possible equations a student might solve as well as actions she might take. Thus, developing and testing a Bayesian inverse planning model for algebraic equation solving addresses the question of whether this type of model is appropriate for interpreting learners' behaviors in environments that may require many individual problem-solving actions.

Through a series of three experiments, we explore the feasibility and effectiveness of applying a Bayesian inverse planning model to algebraic equation solving. In a fourth study, we extend the framework to demonstrate how to adapt existing production rule models of mathematical ability into the inverse planning model context. Experiment 1 demonstrates that the model can interpret people's equation solving, and it shows that for individual equation solving steps, the model's inferences about people's actions are consistent with the inferences made by human annotators. By varying whether the interface provides immediate correction to learners on every step of equation solving, we show that such correction is not necessary for the model to interpret equation solving effectively and that some learners may become frustrated by this type of correction, suggesting the need for systems that permit more naturalistic problem solving. Since Experiment 1 does not address the model's overall accuracy for assessing individual learners, in Experiment 2 we address this question by comparing the model's inferences about learners' skills to inferences about the same learners by experienced mathematics teachers, showing that model is generally consistent with teachers and adds value over a simpler measure of number of problems correct. These first two experiments provide evidence for the accuracy of the inverse planning model's assessments, but they do not address if that accuracy is sufficient for personalizing instruction for learners. In Experiment 3, we demonstrate how the model's inferences can be used to choose what feedback will be most helpful for an individual learner and quantify the benefits of choosing feedback based on the model.

Each of the first three experiments focuses on the domain of equation solving, and applying inverse planning to this domain required creating a model of the possible misunderstandings people might have about equation solving. To demonstrate how inverse planning can be applied to other mathematical domains without needing to create this model of misunderstandings from scratch, we end by showing how to transform an existing model of student's misunderstandings about fraction arithmetic into the form required for inverse planning. In Experiment 4, we use the new inverse planning model to assess people's misunderstandings about fraction arithmetic, demonstrating both that its results are consistent with hand-coding of student strategies and that it can handle a larger number of possible misunderstandings than was addressed in equation solving. Together, these four experiments provide evidence for the feasibility of using Bayesian inverse planning in mathematical domains, both to make sense of students' understanding in a fine-grained way and to modify instruction based on these inferences about understanding.

In the remainder of the paper, we begin by discussing related literature in the learning sciences and cognitive science on recognizing learners' understanding of algebra. We then describe the Bayesian inverse planning model we developed for making inferences about learners' misunderstandings about equation solving. We next turn to the series of experiments detailed above, exploring how well this model captures learners' problem solving as well as how the model can be used to personalize learners' experiences in an educational technology. We end by discussing the limitations of this work as well as future steps for equation solving specifically and inverse planning models more broadly.

2. Related literature

There is significant work in both education and cognitive science on both modeling students' algebra knowledge and on the use of Bayesian networks for assessments. We first focus primarily on models related to algebraic equation solving and then provide a short discussion of existing Bayesian methods for student assessment, focusing on how these differ from the proposed Bayesian inverse planning model.

2.1. Existing models of algebra understanding

There has been a great deal of interest in modeling students' algebra knowledge and categorizing student errors, both in education and in cognitive science. One popular automated system is Cognitive Tutor Algebra, which improves students' algebra skills and standardized test scores (Koedinger et al., 1997). This system uses knowledge tracing (Corbett & Anderson, 1995) to track discrete student skills, and it typically structures problem solving into individual steps which students must complete correctly before continuing. Another tutor, Aplusix, allows students to solve equations in a more freeform manner, but it does not naturally maintain a model of knowledge across problems (Nicaud, Chaachoua, & Bittar, 2006). ALEKS (Falmagne, Cosyn, Doignon, & Thiéry, 2006) is commonly used for remedial mathematics practice in college settings and relies on Knowledge Space Theory (Falmagne & Doignon, 2011) for modeling what types of problems students will be able to solve. While relying on sophisticated testing methods to recognize patterns associated with different gaps in understanding, this system uses only students' final answers to infer their understanding. ASSISTments provides students with algebra practice and guidance, and employs a detailed cognitive model for tracking skills (Razzaq, Heffernan, Feng, & Pardos, 2007). Inverse planning adds to these existing models using free-form problem solving to recognize misunderstandings without requiring correctness of individual steps. Unlike alternative models, it looks both at the actions on each individual step and the pattern of action choices indicating students' strategy when solving problems. Examination of learners' strategies has been more commonly used in open-ended learning environments where learners make choices about what simulations to conduct or how to interact with virtual agents (i.e., Gobert, Sao Pedro, Raziuddin, & Baker, 2013; Kinnebrew, Loretz, & Biswas, 2013).

Cognitive science models have also been proposed to describe the errors that occur in algebraic problem solving. Koedinger and MacLaren (2002) created a detailed process model for how students solve algebra problems, including "buggy" rules that produce particular errors. The model both provided significant coverage of actual student data and buggy rules with plausible methods of acquisition. Brown and VanLehn (1980) propose a theory of errors in which students "patch" partial knowledge with erroneous actions.

Research has also focused on categorizing patterns of algebra errors and specifying the types of buggy rules that occur (e.g., Payne & Squibb, 1990; Sleeman, 1984). Some work identified a limited set of "mal-rules" that could explain most students' solutions, although these rules were not always applied consistently (Payne & Squibb, 1990; Sleeman, 1984). While later work suggests that explicitly teaching that these mal-rules are incorrect is no better than simply reteaching material (Sleeman, Kelly, Martinak, Ward, & Moore, 1989), detection of mal-rules and separation of types of errors might be helpful for guiding what material to reteach. The inverse planning model we propose builds on these investigations of mal-rules, linking them into a statistical framework to allow us to automatically detect when learners use them.

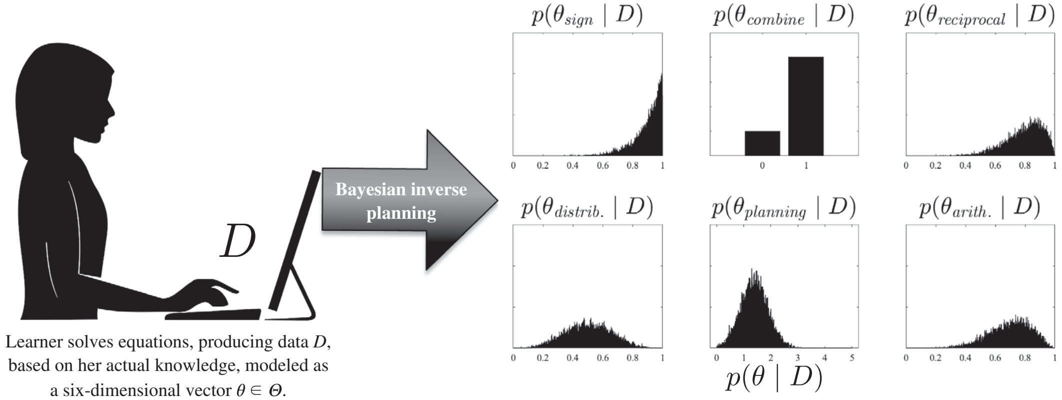
2.2. *Bayesian models for student assessment*

Bayesian models for student assessment have a rich and successful history in making assessments about individual student skills. For example, as previously noted, Bayesian knowledge tracing (Corbett & Anderson, 1995) is used throughout the successful Cognitive Tutor platforms (Ritter, Anderson, Koedinger, & Corbett, 2007), and a number of extensions have led to it being applicable in a wide variety of circumstances. For example, conjunctive knowledge tracing extends the model to allow for application of multiple skills in a single step (Koedinger, Pavlik Jr., Stamper, Nixon, & Ritter, 2011), and Feature Aware Student knowledge Tracing (FAST) provides a way to incorporate features into knowledge tracing as an alternative way to model multiple subskills and blend the model with other psychometric approaches like item-response theory (González-Brenes, Huang, & Brusilovsky, 2014). The inverse planning model differs from these approaches through its inclusion of a model of strategic planning that presumes students choose actions based on their goals, placing a probability distribution over their action choices and naturally modeling how changes in incentives impact student choices.

Outside of knowledge tracing, Bayes' nets have been used to make assessments about student skills in a variety of environments. These approaches typically provide greater flexibility in the connections across skills and opportunities to demonstrate knowledge, at the cost of increased complexity. For example, Conati, Gertner, and VanLehn (2002) developed a Bayes' net model of students' physics understanding. This model did allow for some plan recognition, enabling tracing of student skills based on their choices, but the plans were not assumed to be strategic. Specifically, students' use of different production rules was based on whether underlying skills were mastered or unmastered, but the approach did not incorporate student responses to incentives in the model. Inverse planning takes a similar strategy of assuming a (possibly complex) Bayes' net relates student knowledge and actions, and augments it with the assumption that students' plans are multi-step and sensitive to different reward structures. The strategy in Conati et al. (2002) is the most similar to our work among the literature on applying Bayes' nets to student assessment, but there also exists work less focused on planning that uses Bayes' nets for this purpose. For instance, Kim, Almond, and Shute (2016) draw on ideas from evidence-center design to make assessments about students' skills based on their actions in a physics game. This work uses a Bayes' net to model the covariance among student behavior and underlying skills, but unlike inverse planning, the Bayes' net is not developed as a generative model relating student knowledge to specific actions. Creating such a generative model makes it easier to incorporate planning and to generalize the same model across different formats of tasks and problems.

3. Inverse planning

To assess a learner's equation solving proficiency through observations of her behavior, we used a Bayesian inverse planning approach (Rafferty et al., 2015). Bayesian inverse planning takes as input a set of observed behaviors from a learner and outputs a posterior distribution over possible different levels of knowledge or ways of understanding (see Fig. 1 for an overview). The observed behaviors here are all steps created by a learner when she is asked to solve an equation; we refer to these behaviors as a "sequence of steps" to solve an equation. Inverse planning assesses proficiency based both on the individual behaviors a learner takes (e.g., transforming $3x = 4 + 6$ into $3x = 10$), and the sequence of those behaviors, accounting for the fact that a learner has many possible choices about how to solve a problem and eliminating the need to constrain the learner's choices about what steps to take. Bayesian inverse planning is based on a generative model: It models how likely a person would be to choose each possible solution step if she had a particular understanding of algebra, and then uses this model to infer what understanding is most likely to have resulted in the observed sequences of


D
Bayesian inverse planning
$p(\theta_{sign} \mid D)$
$p(\theta_{combine} \mid D)$
$p(\theta_{reciprocal} \mid D)$
$p(\theta_{distrib.} \mid D)$
$p(\theta_{planning} \mid D)$
$p(\theta_{arith.} \mid D)$
0 0.2 0.4 0.6 0.8 1
0 1
0 1 2 3 4 5
$p(\theta \mid D)$

Learner solves equations, producing data D, based on her actual knowledge, modeled as a six-dimensional vector $\theta \in \Theta$.

Bayesian inverse planning processes the data D and outputs an assessment of the learner based on the data. The assessment is a distribution over all possible vectors $\theta \in \Theta$.

Fig. 1. Modeling learner's knowledge using inverse planning. The dimensions of θs refer to proficiency on different aspects of solving algebraic equations, such as *planning* how to sequence steps or changing the *sign* of a term when moving it from one side of the equation to another.

solution steps. To create this generative model, we must specify the representation of possible understandings and the generative model of how a person chooses steps based on her understanding. We can then describe how to generate a probability distribution over possible understandings given the observed solutions.

3.1. Representation of understandings

To make fine-grained inferences about specific ways of understanding equation solving, we model understanding as the level of proficiency for six different skills. For each skill, the proficiency indicates whether the person generally applies the skill correctly or if she makes a particular type of error. The different levels of proficiencies form a *hypothesis space* Θ of possible algebra understandings. The hypothesis space is based on past education and psychology research, and each understanding in this space is a six-dimensional vector θ reflecting a possible level of proficiency for each skill.

Four dimensions of each θ are skills related to specific equation manipulation actions and are based on mal-rules from prior work (Payne & Squibb, 1990; Sleeman, 1984). Since most mal-rules are associated with specific actions, we describe these mal-rules and the related skills in the next section (with the description of allowed actions), although we note here that three are represented as probabilities of applying a skill correctly and the fourth is a binary parameter representing whether particular actions are included in the space of actions the solver considers.

Two additional dimensions of θ affect each action choice and transformation. The *arithmetic* parameter is the probability of *not* making an arithmetic error in each operation in a transformation. For instance, when distributing the coefficient in $3(x + 2 + 3)$, there are three opportunities to make an arithmetic error. This parameter separates misunderstandings about the rules of algebra from difficulties with arithmetic. Because our focus is primarily on localizing what area of equation solving a person may be struggling with, the arithmetic error parameter does not differentiate the probability of making an error on different types of arithmetic operations or different numbers, even though it is likely that these characteristics impact error probability for individuals. Incorporating a finer-grained model, such as one that makes arithmetic error probability dependent on the magnitude of the numbers involved, would be easy to do given the modular nature of inverse planning. The final parameter, which we will refer to as the *planning* parameter, relates to solving efficiency. More efficient solvers tend to take fewer steps to move from an unsolved equation to a solved equation, and typically do not give up and leave an equation unsolved. Less efficient planning could be due to inability to identify the most efficient solution or to a solver's assessment that she may struggle to carry out the most efficient solution (e.g., because it would require more cognitive load due to more complex arithmetic operations).

In summary, all six dimensions of θ reflect the learner's proficiency on one particular type of equation solving skill. For all dimensions, larger values indicate more normative understanding. The four parameters involving mal-rules and the *arithmetic error* parameter have a maximum value of one while the *planning* parameter may be any number ≥ 0.

3.2. Generative model of equation solving

In an inverse planning model, we must define a generative model of equation solving that formally describes our assumptions about how people choose actions based on their understanding. Our model relies on treating algebraic equation solving as a Markov decision process (MDP), a common decision-theoretic model for stochastic environments involving agents that are trying to achieve some goal or maximize some reward (see Sutton & Barto, 1998, for an overview of MDPs). In equation solving, the agent is a person who is choosing actions to try to achieve her goal of solving an equation with as few steps as possible. With each action, the person moves from one (partially solved) equation to another; see Fig. 2 for an overview of the model. Formally, an MDP is defined as a tuple ⟨(*S*)*tates*, (*A*)*ctions*, (*T*)*ransitions*, (*R*)*ewards*, discount factor γ⟩. The set of states S are the possible situations in which a learner might be choosing an action. We define each equation as a state, reflecting the fact that a learner chooses what action to take next based on the current equation. The set of actions A are the ways that the learner might transform one equation to another. We consider six general types of actions, corresponding to possible equation transformations:

1. **move:** Move a specific term from one side of the equation to another.
2. **divide:** Divide both sides by the coefficient of a specific term.
3. **multiply**: Multiply both sides by a specific constant.
4. **combine:** Combine two specific terms on the same side of the equation.
5. **distribute:** Multiply out a specific complex term like $3(x + 6)$ to get $3x + 18$. This action may also include distributing only part of the coefficient, such as transforming $- 3(x + 6)$ to $-(3x + 18)$.
6. **stop:** Stop solving the current problem.

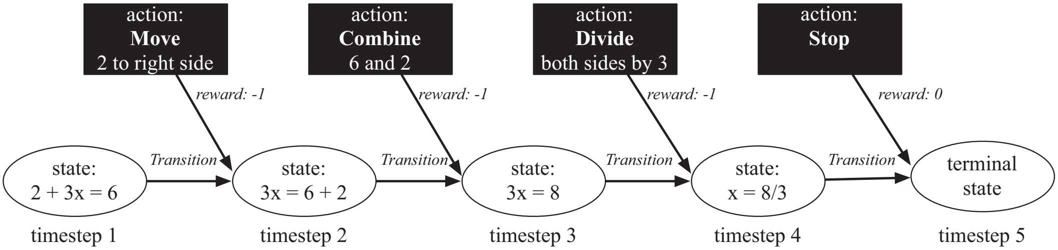


Fig. 2. Modeling equation solving as an Markov decision process (MDP). Each equation is a state, and at each timestep, the learner chooses an action and generates the next state (equation). The *transition* model encodes the probability of possible next states: Different next states will have different probabilities based on the combination of the learner's knowledge (θ), her action choice, and the current state. As shown in the transition from timestep 1 to timestep 2, transitions with errors in arithmetic or the algebra rules are included in the model, with probability dependent on the learner's skills (θ). At each time step, the reward may vary based on the state and action; here, there is a negative reward for all actions except choosing **stop** when the equation is in a solved state.

The first five actions represent possible ways to manipulate an equation; below, we describe how errors in these actions are modeled. The final action is taken by a learner who believes she has finished solving the problem or who is giving up.

For a specific equation, there will be many instantiations of each of these general actions. For example, if the current state was $3 + 2 + 5x = 10x$, there would be four move actions: move 3, move 2, move $5x$, and move $10x$. Each of these is a separate action in the MDP. The θ_{combine} parameter impacts which actions the learner considers. If this binary parameter is equal to 1, then the leaner is assumed to only consider **combine** actions that combine two variables or two constants; here, "**combine** 3 and 2" would be the only action considered. If this parameter is equal to 0, then the learner considers combining all pairs of terms on each side.[2] This would result in three combine actions for this example, adding "**combine** 3 and $5x$" and "**combine** 2 and $5x$."

The transition model T of the MDP represents the probability of each possible next state (equation) given the current state and action. In normative equation solving, each state and action would have a deterministic next state; for instance, if the current state was $2x + 5x = 14$ and the action the learner took was "**combine** $2x$ and $5x$," the next state would be $7x = 14$. However, we model the state transitions probabilistically based on the learner's hypothesis θ.

Three dimensions of θ affect the transition model: θsign, $\theta_{\text{reciprocal}}$, and $\theta_{\text{distributive}}$. Each of these is related to a mal-rule reported in prior work (Payne & Squibb, 1990; Sleeman, 1984), and the value of each of these dimensions is a probability that represents the probability of *not* using related mal-rule when executing the relevant action. The θ_{sign} dimension is related to the *sign* error mal-rule, which refers to moving a term from one side of the equation to the other without changing the sign (e.g., $2x + 3 = 6 \rightarrow 2x = 6 + 3$). If the learner is making a **move** action, then with probability equal to the *sign* parameter, she will execute this action correctly, and with probability equal to one minus the *sign* parameter, she will fail to change the sign of the term she is moving. For instance, if $\theta_{\text{sign}} = 0.75$, then a learner moving 8 in the equation $8 + x = 16$ will write down the next equation as $x = 16 - 8$ three-quarters of the time and write down the next equation as $x = 16 + 8$ one-quarter of the time. The $\theta_{\text{reciprocal}}$ dimension is related to the *reciprocal* error mal-rule, which refers to multiplying rather than dividing by a coefficient (e.g., $5x = 1 \rightarrow x = 5$). $\theta_{\text{reciprocal}}$ also a probability, representing whether the learner divides by the coefficient or its reciprocal in **divide actions**. The $\theta_{\text{distributive}}$ dimension is related to the *reciprocal* error mal-rule. In this mal-rule, solvers multiply only the first term in parentheses by a value (e.g., $4(x + 3) \rightarrow 4x + 3$). This mal-rule is relevant to transitions for the **distribute** action.

The final parts of the MDP are the reward model R and the discount factor γ. The reward model encodes the goals of the actor (in this case, the learner). We include a small negative reward for each action and a large negative reward for choosing **stop** when the equation is in an unsolved state. This corresponds to modeling learners as trying to solve problems and trying to do so using as few steps as possible. The discount factor γ encodes the relative value of immediate versus future rewards, and it is needed in some MDPs where the total possible reward is infinite. The discount factor γ could be tuned to people's sensitivity for immediate versus longer-term rewards, but in the equation solving setting where the total number of steps is relatively small, we represent learners as valuing immediate and future rewards equally by setting $\gamma = 1$.

Now that we have defined the dynamics of the MDP given the learner's understanding θ, we must model how the learner chooses an action given the current equation; this is known as the learner's *policy*. Consistent with other work using MDPs to model human action solving (Baker, Saxe, & Tenenbaum, 2009; Rafferty et al., 2015), we model people as following a noisily optimal policy when choosing an action a for a state s that corresponds to the current equation:

$$P(a/s) \propto \exp(\theta_\beta \cdot Q_\theta(s,a)), \tag{1}$$

where Q_θ (s, a) is the long-term value of being in state s and taking action a and θ_β is the planning parameter. In an MDP, the long-term value (Q-value) of being in state s and taking action a is equal to the expected sum of the immediate reward of taking action a in state s and the rewards obtained on all future steps. Here, we denote this as Q_θ (s, a), with the subscript θ indicates that this value is dependent on the person's understanding of algebra, since that understanding may change what actions she believes are possible or what next equation she generates from the current equation. The planning parameter θ_β influences action choices by modeling how consistently people choose actions with highest value. If θ_β is equal to zero, the person is modeled as choosing actions uniformly at random, while very large θ_β means the person almost always chooses the action with the highest expected long-term value. Intuitively, this policy assumes people tend to choose actions that they think will help them solve the problem efficiently, but they do not do so deterministically.

In an MDP with discrete state and action spaces, the Q-values can be calculated using value iteration, a dynamic programming algorithm (Bellman, 1957). A technical difficulty arises in our application because the state space S is infinite. To handle this complication, we rely on the fact that for equations that have similar structures, the value of particular types of actions is similar. For example, for equations $3 + 7x = 5$ and $10 + 20x = 15$, the value of moving 3 to the right side in the first equation should be similar to the value of moving 10 to the right side in the second equation. To calculate the Q_θ $(s_, a)$ values for each state-action pair, we thus aggregate over states, representing all equations with the same number of variables, constants, and "complex" terms on each side as the same "macro-state." A complex term is a term involving the distributive property, such as 4 $(x + 3)$. This is a simplifying assumption in two ways: From a cognitive science perspective, it assumes that learners are relatively insensitive to differences in specific numbers and differences in ordering of terms, and from a computational perspective, it assumes that all aggregated states will have the same values for each action. We recognize that this model ignores real differences in the way learners work with different numbers, just as the unidimensionality of $\theta_{\text{arithmetic}}$ to represent all arithmetic errors ignores these differences; we hope that future work will create finer-grained models of errors, and we note that the overall framework can be applied to making inferences about finer-grained models. With respect to the computational perspective, the definition of θ and the state transitions means that this aggregation does not lead to much approximation error because in most cases, all grouped states in the same macro-state do have similar values when paired with equivalent actions. We provide more details about the value iteration algorithm and our application of it in this setting in Appendix A.

3.3. Assessing understanding from actions

Given a generative model of equation solving and a space of possible hypotheses Θ representing different understandings of algebraic equation solving, we calculate a distribution over Θ given a learner's observed problem solutions D using Bayes' rule: for each $\theta \in \Theta$,

$$p(\theta|D) \propto p(\theta)\prod_{d \in D} p(d|\theta), \tag{2}$$

where $p(\theta)$ is the prior distribution over the hypothesis space and each $d \in D$ is the step-by-step-solution to a single equation. While many possible prior distributions are possible, we use a prior distribution that weakly favors higher levels of proficiency (see Appendix C for details). This means that the algorithm favors the part of the hypothesis space indicating normative algebra understanding unless it observes evidence of non-normative steps. The likelihood $p(d|\theta)$ represents the probability the person would produce the observed step-by-step solution if θ reflected her true understanding of equation solving. This term can be calculated using the MDP defined by θ, but one technical difficulty is that the actions the learner takes are unobserved: We only see the equations the learner writes down when solving. We thus marginalize over possible actions (see Appendix A for details), and we can extend this procedure to cases where a learner takes multiple actions without writing down the intermediate step. For instance, a learner might write $x + 5x = 12 + 6$ and then $6x = 18$; no one action explains this transformation, so the model will consider possibilities for the omitted step (e.g., $6x = 12 + 6$ or $x + 5x = 18$) when computing the likelihood.

Because the hypothesis space Θ is continuous, the posterior distribution cannot be calculated exactly. Instead, Markov chain Monte Carlo (MCMC) methods, specifically the Metropolis–Hastings algorithm, are used to compute an approximate posterior distribution. Each step in the algorithm samples some $\theta \in \Theta$, and that θ affects the value of the posterior distribution of the sample; four dimensions of θ affect the likelihood through the transition model, one dimension affects the likelihood through the space of actions considered (including combining unlike terms or not), and the final dimension affects the probabilities of each action choice. The collection of accepted sampled points is an approximation of the posterior distribution. See Appendix C for more details about the sampling.

Fig. 3 shows the inferred posterior distribution for a single experiment participant. Each individual plot shows the approximate posterior for one dimension of θ. The

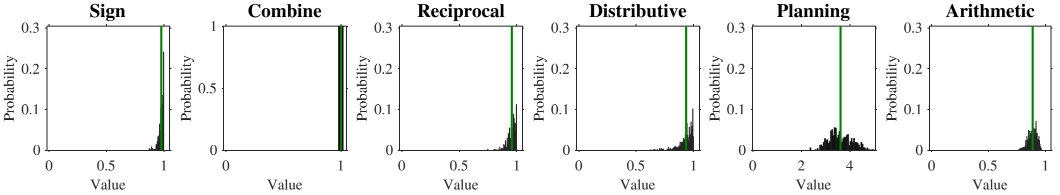


Fig. 3. The inverse planning algorithm's assessment for a learner from Experiment 3. Each plot shows the posterior distribution for one dimension of θ, marginalizing over the other dimensions. Larger values are closer to mastery, and vertical green lines show the mean value of the posterior.

posterior distribution indicates both the most likely proficiency for each skill (highest individual bar) and the algorithm's confidence, indicated by how large the range of values is that has non-negligible probability; the vertical green lines indicate the mean value of the posterior, which can provide a point estimate for each dimension. In the figure, both the parameter for moving terms and the distributive property are close to one, but the estimate for moving terms is more concentrated, meaning the algorithm is more certain of the value for this parameter. There is a lower estimated proficiency for arithmetic than for the other skills. The planning parameter is relatively large, indicating that this learner chooses her actions efficiently most of the time and does not give up on unsolved equations.

4. Experiment 1: Evaluating freeform equation solving

The Bayesian inverse planning model that we have presented goes beyond prior applications of inverse planning by interpreting much less constrained data. Rather than examining people's choices in games where there is a fixed set of options (Rafferty et al., 2015; Rafferty, Zaharia, & Griffiths, 2014), the application to equation solving attempts to model possible actions out of an unlimited set of possibilities. Thus, we first test the feasibility of using the Bayesian inverse planning model to assess people's understanding by collecting step-by-step equation solving data and interpreting it using this model. Many existing tutors structure problem solving to provide step-by-step feedback, which could make the interpretation task easier by constraining the possible equation solving paths. While Bayesian inverse planning does not require such structuring, it is possible that such structuring would allow it to interpret more solutions. Thus, we conducted an experiment comparing a structured interface in which people had to enter each step of an equation correctly prior to continuing to the next step with an unstructured interface in which people were asked to enter each step, but could combine steps or include erroneous steps. We focused on adult problem solvers who had some algebra experience, but may not have used their skills recently. These solvers might be particularly likely to have gaps or errors in their understanding, and might appreciate the chance to solve the equations without immediate correction to get the chance to show what parts of equation solving they do know, rather than being stopped repeatedly prior to that point. Thus, as a secondary goal, we also examined how the choice of interface affected people's satisfaction with the system and their learning outcomes.

4.1. Methods

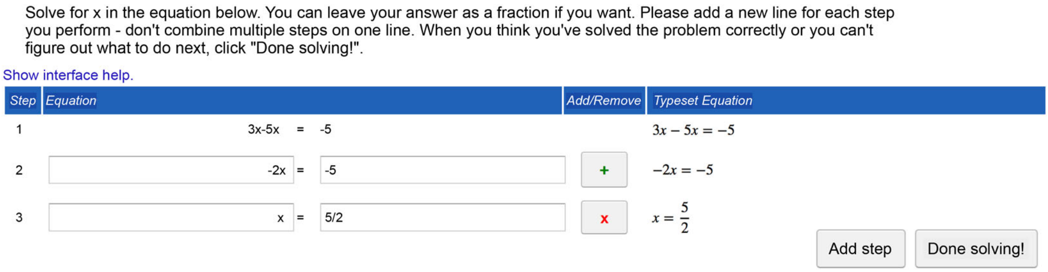
4.1.1. Participants

A total of 40 participants in the United States were recruited from Amazon's Mechanical Turk (AMT) and compensated \$3 for session 1, \$5 for session 2, and \$7 for session 3. Participants were asked to affirm that they had experience with algebra, either in a secondary or postsecondary class, and had not completed college math classes beyond algebra. We continued recruiting participants until we had 40 who completed all three sessions.

4.1.2. Stimuli

Across the three sessions, participants completed a multiple-choice assessment (sessions 1 and 3), solved algebra problems on a website (session 2), and responded to surveys about their mathematics background and the usability of the website (sessions 1 and 3). The multiple-choice assessment was based on College Board ACCUPLACER® tests used for math placement in many postsecondary institutions (Mattern & Packman, 2000); questions were adapted from sample questions from the College Board (College Board, n.d.). There were 12 "elementary algebra" questions and 16 "college math" questions. Instructions informed participants that if they did not know how to solve a problem, they could leave the question blank.

On the algebra website we developed,[3] participants are shown the equation to solve, and with each problem step, they add a line to show their work (Fig. 4). The last column in the interface updates with the rendered version of the user's input in real time, with the intent of lowering barriers to interpreting typed mathematics. In the *corrected* interface, the current step is checked via a symbolic algebra system (Kramer, 2010–2014) when the add step button is clicked. If the step is mathematically incorrect, combines multiple steps, or is not the best next action, then a dialogue tells the participant to correct the step before continuing. The dialogue distinguishes between syntactic errors (e.g.,


Solve for x in the equation below. You can leave your answer as a fraction if you want. Please add a new line for each step you perform - don't combine multiple steps on one line. When you think you've solved the problem correctly or you can't figure out what to do next, click "Done solving!".

Show interface help.

Step	Equation			Add/Remove	Typeset Equation
1	3x-5x	=	-5		$3x - 5x = -5$
2	-2x	=	-5	+	$-2x = -5$
3	x	=	5/2	×	$x = \frac{5}{2}$

Fig. 4. Screenshot of the step-by-step problem-solving interface on the website. The first line in the interface shows the problem the user must solve, and the user adds lines to show her problem-solving steps.

having an extra close parenthesis), mathematical errors (e.g., making an arithmetic error), and action choice errors (e.g., moving a constant term as the next step in $7x = 2 + 4$). Allowed action choices are calculated by comparing the user's actions to all possible legal actions. We allow any action that is close to optimal according to the generative model of equation solving (see Appendix B for details).

In the *freeform* interface, the system checks for syntactic errors when the user submits the entire set of steps for a problem and highlights rows with syntactic errors. After the user either corrects syntactic errors or moves forward without correction, steps with mathematical errors are highlighted for the user to review. This results in similar amounts of feedback across conditions. The interface moves to the next problem when the user clicks a button to indicate she is done reviewing.

The mathematical background survey included four questions about mathematics background and postsecondary education and two questions about age and gender. In the usability survey, all participants completed 10 questions on a 5-point Likert-scale, with participants in the *corrected* condition completing two additional questions. The initial 10 questions were adapted from Lund (2001)'s usability survey and focused on perceived learning as well as satisfaction with website features. The *corrected*-only questions focused on feelings about having to get each step correct before continuing and whether they found this feedback helpful. The survey also contained three open-ended questions asking for suggestions for improvement and what features were frustrating.

4.1.3. Procedure

Participants completed three sessions, separated by at least 1 day. To encourage participants to participate in all three sessions, participants were told of the escalating pay scale in the first session and asked to participate in the task only if they intended to participate in all three sessions. At the beginning of each day after they completed one of the first two sessions, participants were sent an email providing a link to the task for the next session.

The first session included the multiple-choice assessment followed by the mathematical background survey. In the second session, participants were randomly assigned to one of two conditions and solved 20 problems on the algebra website. The website included a short tutorial about how to use the interface. This tutorial was identical across conditions except for explanations of differing features: the step-by-step error checking in the *corrected* condition, and the opportunity to review the problem after submission in the *freeform* condition. The problems to solve were automatically generated based on templates (see Appendix D for details). All participants completed two problems based on each of the same 10 templates in a randomized order. In the final session, participants completed the same assessment as in session one, and then responded to the usability survey.

4.2. Results

The vast majority of participants who began session two, the session with the experimental manipulation, completed all three sessions: Only one participant who completed more than one problem in session two did not complete the third session. This participant was in the *corrected* condition. Two other participants from each of the two conditions did not complete the third session. Each of these participants completed one or zero problems in session 2. In the remainder of this section, we analyze the results of those 40 participants who completed all three sessions.

Overall, participants in both conditions attempted similar numbers of problems and both answered about 50% of problems correctly on the website, indicating that they were far below ceiling. More detailed information about improvement from pretest to posttest as well as our secondary goal of examining participants' satisfaction with the website is available in Appendix E.

We applied Bayesian inverse planning to assess the understanding of each participant, and then performed a number of analyses to determine how consistent the results were with human inferences about the participant's equation solving, how consistent simulations from the model were with actual observed behavior, and whether there were differences in model performance between the correct and freeform conditions.

First, we assessed the algorithm's performance based on its ability to interpret step transitions as actions in the model. We initially analyzed participants' results under the assumption that participants do not combine actions; that is, each two-step sequence should be the result of carrying out a single action. Fig. 5b shows the proportion of two-step sequences for which at least one action had non-zero probability; this includes all attempts at steps, including steps that participants entered in the *corrected* condition that were incorrect or the wrong action. When each step transition must be interpreted as a single action, many more steps in the *corrected* condition can be interpreted than in the *freeform* condition. One cause of this issue may be that our model of what constitutes a single step is not always consistent with participants' beliefs about what defines a step; as discussed in the "Assessing understanding from actions" section, the model can interpret combined actions if we sum over sequences of multiple actions rather than only a single action. As Fig. 1c shows, this results in similar coverage across conditions, with over 90% of two-step sequences interpreted by the model. Under the assumption of permitting

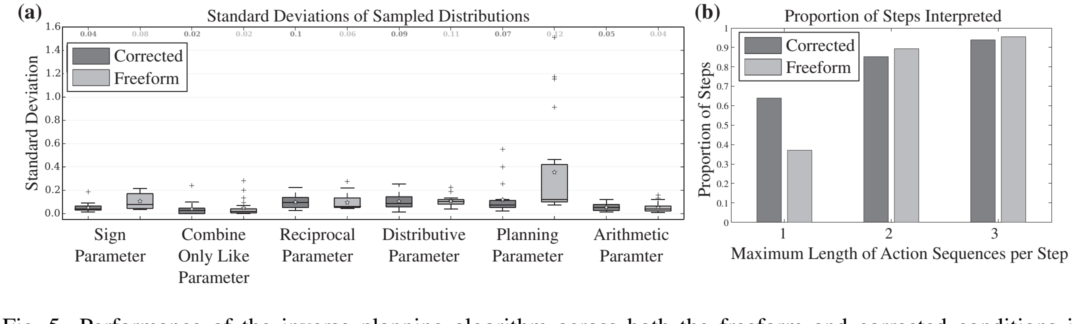


Fig. 5. Performance of the inverse planning algorithm across both the freeform and corrected conditions in Experiment 1. (a) Distribution of the standard deviations of the sampled distributions, by condition. (b) Proportion of steps where inverse planning found at least one valid action sequence.

two-step sequences of actions, the MCMC sampler exhibited good mixing behavior for the majority of participants, as measured by R; see Appendix F for details.

Measuring action coverage indicates whether the model can find any interpretation of an equation solving step, but it does not address whether the model's interpretations are correct. Human interpretations can provide a gold standard for what action a person was taking when transforming one equation to another, and agreement between people when interpreting actions is a way to determine the maximum accuracy we might expect a system to achieve. A subset of 148 problems (a total of 582 actions) with at least one mathematically incorrect transformation were randomly selected for annotation; annotation was restricted to those that included a mathematically incorrect transformation to ensure that our measure of accuracy included both correct and incorrect transformations. The step-by-step problem solutions were combined with the algorithm's inferences about what actions could have accounted for each step. Two undergraduate research assistants were chosen to annotate the data, and they were trained in how to interpret the system's output and evaluate whether a step was correctly interpreted by the algorithm, meaning that one of the actions given by the algorithm was consistent with the annotator's judgment about the true action. Steps that the algorithm could not interpret were always marked as inconsistent with the annotator's judgment, even if the annotator also had difficulty interpreting the person's action. After coding several sample problems together with the first author, the annotators independently evaluated the problems. They agreed on 92% of actions, with good inter-annotator agreement (Cohen's $\kappa = 0.73$); disagreements were reconciled via discussion after all initial coding was complete. After reconciling disagreements, the annotators found the algorithm's interpretations to be consistent with their own for 81% of the actions.

We next used posterior predictive sampling to compare characteristics of people's actual behavior with that which would be expected based on the model's inferences. Specifically, for each participant, we sampled from the model's inferred posterior distribution 1,000 times. For each of these 1,000 samples, we simulated the model solving each equation that the participant solved and recorded the number of steps taken on each equation, the number of problems correct, the correctness rate on individual steps (i.e., what proportion of steps were mathematically consistent with the preceding step), and the number of problems where a numeric answer was computed prior to the "stop solving" action. The correlation between each of these measures for the simulated data from the model and the actual participant data is an assessment of how accurate the model is at capturing real equation solving behavior. For three of these four measures, there was a significant correlation between the results for participants and the results from simulating the model (number of problems correct: $r(38) = .81$, $p < .001$; correctness rate on individual steps: $r(38) = .70$, $p < .001$; number of problems where an answer was computed: $r(38) = .97$, $p < .001$). Number of steps taken on an equation was not significantly correlated between the model simulations and people's actual data ($r(38) = .24$, $p = .14$). One explanation for this lack of correlation is that for the corrected steps condition, participants often indicated they were done with a problem if they took a step that corresponded to an incorrect action, rather than trying other actions or trying the same action again. The model was more likely than participants to try the same action multiple times, and the noise model did not selectively add weight to the stop action, as it seemed that people did. Overall, though, this suggests that the model is a reasonable fit for people's equation solving.

Finally, we explored whether the algorithm's assessments were as confident based on the *freeform* data as from the *corrected* data. While interpreting *freeform* data is a natural fit for this algorithm, these data are also likely to exhibit ambiguity: Since the interface does not force particular action choices and allows mathematical errors, there may be cases where more than one action or error could account for a step, and people may combine actions.

To measure how confident the skill-by-skill assessments were in each condition, we calculated the standard deviation of the sampled values for each participant. As shown in Fig. 5a, the distribution of these standard deviations is generally similar across conditions except for the planning parameter: For most parameters, the lack of constraints in the *freeform* interface did not decrease confidence. For the planning parameter, we believe the difference in confidence is due to participants in the *corrected* condition giving up more frequently. For example, one participant in this condition saw the equation $-x + 6 = 2 + 2x$ and followed with the step $-x + 6 = 4x$. This is not a mathematically valid next step, so the interface prevented the user from continuing until the step was corrected; this participant eventually submitted the problem without making any further progress. Giving up frequently results in a lower planning parameter, and thus participants in the *corrected* condition had lower inferred planning parameters on average. Since variance in sampled values tends to increase with the parameter's magnitude, we would expect that this difference in values would result in less absolute confidence for this parameter in the *freeform* condition. This difference also highlights the fact that one might want to separate choices about when to stop solving from suboptimal choices among problem-solving actions; this could be incorporated into the algorithm by separating the choice function into two parts, one of which decides between stopping and acting and the other decides among actions.

Overall, these modeling results suggest that problem solving in the more *freeform* interface is as interpretable by the model as problem solving in the *corrected* interface, and that in both cases, the model can interpret almost all equation transformations and these interpretations are consistent with human interpretations.

4.3. Comparison to alternative models

The inverse planning model is relatively complex, combining a fine-grained domain model that allows differing levels of proficiency across skills with the assumption of noisily optimal action selection that is sensitive to the long-term value of different action choices. To determine whether this complexity is necessary and how these different components contribute to the model's overall success in making inferences from equation solving behavior, we consider two ablated versions of the inverse planning model: The *single-proficiency* model assumes there is a single proficiency level for each person that applies to all of their skills, although noisily optimal action selection is maintained, and the *uniform-action* model assumes that actions are selected uniformly from the possible actions, although different proficiencies are permitted for different skills. We fit the data for each participant to both of these alternative models.[4] For each participant, we then determined whether the inverse planning model was a better fit to their data than each ablated model by performing a χ^2 goodness-of-fit test, resulting in model fit tests for each participant. If some participants are better fit by the inverse planning model rather than an ablated model, it suggests that the ablated component is in fact necessary to model at least some nuances of people's equation solving.

Equation solving data for a sizable minority of participants (13/40) were better fit by the inverse planning model than by the *single-proficiency* model ($\chi^2(4) > 9.49$ for each of these 13 participants). The fact that the inverse planning model is not always a significantly better fit than the *single-proficiency* model may be attributable to several factors. First, we do expect correlations across these skills for at least some participants. Consider participants who answer almost all questions correctly. The inverse planning model would infer that they are close to proficient for almost all skills, and thus the extra expressivity of multiple parameters is not needed; we return to this point in Experiment 3 when considering how to use the proficiency estimates to make decisions about instruction. Second, limited equation solving data are likely to mean that the *single-proficiency* model performs better here than if data were collected over a longer assessment. Since evidence for proficiency on some skills is demonstrated only rarely, such as whether one correctly distributes a coefficient over all terms, small differences in parameter values are not detectable. Thus, the fact that 32.5% of participants are better fit by the inverse planning model is suggestive of people's understanding being truly multi-faceted, as is supported by other models of equation solving that do not take planning into account (e.g., Pane, Griffin, McCaffrey, & Karam, 2014).

For 87.5% of participants (35/40), the inverse planning model was a better fit for their data than the *uniform-action* model ($\chi^2(1) > 3.84$ for each of these 35 participants). Thus, while modeling planning increases the complexity of the model, this appears to be an important part of the inverse planning model's ability to account for people's equation solving behavior.

5. Experiment 2: Evaluating algorithm accuracy

As shown in Experiment 1, inverse planning can automatically provide a more fine-grained measurement of someone's algebra performance than a score based solely on correctness. In a second experiment, we assessed the overall accuracy of the algorithm's inferences by comparing its output to teachers' assessments about the same individuals. Teachers are skilled at evaluating individual students' work and identifying their underlying misunderstandings, but teachers also have very limited time to evaluate a large number of students. In light of this, one potential use of inverse planning is to provide teachers with more information about their students, assuming that the model's inferences are sufficiently accurate. Because teachers have significant experience evaluating student work, this comparison may also be more relevant than comparing the algorithm's inferences about individual actions to those made by the annotators in Experiment 1.

In this experiment, we used teacher judgments as the ideal measure of a student's ability on each of the six skills described earlier. We compared the inverse planning algorithm's output to teachers' ratings of participants' proficiency, and we examined how the information provided by inverse planning compares to that provided by assessing only number correct.

5.1. Methods

5.1.1. Participants: Equation solvers

A total of 30 participants in the United States were recruited from AMT and compensated $6. Participants had taken an algebra course and had not completed college math classes beyond algebra.

5.1.2. Participants: Teachers

Three teachers in the New York City metropolitan area gave ratings for all 30 participants. One teacher was a community college mathematics teacher, one was a graduate student with experience teaching elementary and middle school math, and one taught high school math with prior experience teaching freshman college level math.

5.1.3. Stimuli

Each equation solver completed 24 problems in the same algebra website used in Experiment 1, using the freeform interface that allows entering of steps without correction. Problems were generated using the same procedure as in Experiment 1, although 12 rather than 10 templates were each used twice to generate 24 problems instead of 20.

They then filled out a survey where they rated their performance out of 100 on each of six skills. To make the names of the skills more interpretable to participants, we made them slightly more descriptive: *Sign* was called "moving terms," *combine* was "combining like terms," *reciprocal* was "isolating the variable," *distributive* was "distribution," and *arithmetic* and *planning* retained their names.[5] Each skill included a short description (see Appendix D). We also collected demographic information, including gender, age, and a list of math classes they had previously completed.

Teachers were given documents containing each equation solver's work. Each document listed all problems solved, all of the steps taken, and whether each problem was solved correctly. Then teachers were asked to rate each equation solver's proficiency out of 100 for each of the six skills. Like participants, teachers were given the names of the skills and short descriptions (see Appendix D) but not given further instruction about how to decide on their ratings. In addition to scores, we asked teachers to provide explanations about their overall strategy for scoring the participant.

5.1.4. Automatically computing proficiency

The same inverse planning algorithm described above was used to compute assessments for each participant. Examination of the results from Experiment 1 showed that some participants had very low inferred planning parameters because they gave up on some problems. This could be due to inability to plan a solution, or due to other factors such as limited motivation. Low inferred planning parameters tend to lead to noisier estimates of other parameters, since the probability distribution over action choices is more uniform: There is less information about what action the participant is attempting given the context. Because we were not sure how teachers would interpret giving up on problems, we examine two versions of the algorithm: one that includes the choice to stop in inference, and one that does not. The version of the algorithm that does not incorporate the stop action (the “no stop” version) will produce higher planning parameters than the version with stop actions, and it may exhibit greater precision in the other skill estimates. Below, we report results related to both the “stop” and “no stop” versions of the algorithm. Both versions are well-correlated with teacher scores for most skills, but the version of the algorithm that excludes the stop action predicts teacher scores with higher accuracy.

5.2. Results

5.2.1. Inter-teacher and algorithm correlations

Pearson correlations served to evaluate the similarity between the inverse planning algorithm assessment and teacher ratings. Because teachers were given little instruction to standardize their use of the scale, Pearson correlation is a more appropriate measure than interrater reliability, as each teacher may have used the range of the scale somewhat differently. Table 1 shows how correlated each teacher is with the others on all six skills, which serves as an upper bound on how high teacher/algorithm correlations could be. Notably, the combining like terms skill had the lowest set of inter-teacher correlations with none above.[5] Lower correlations across teachers led us to believe that the algorithm’s assessment will also be less correlated with teachers across certain skills.

For most parameters, the algorithm output represents the probability (between zero and one) of using each skill correctly. To compare teacher and algorithm assessments, teacher ratings were converted to z-scores for standardization across teachers, and then for each participant, the mean z-score across teachers was computed for each skill. Positive correlations between the algorithm output for a skill and the mean teacher rating for a skill would indicate that the algorithm’s assessment of student proficiency is consistent with the teachers’ assessments. For this analysis, we use the natural logarithm of the values for the algorithm’s rating of the planning skill, as this parameter is measured on a logarithmic scale within the model. For both versions of inverse planning, correlations between teacher scores and automated assessments were at least moderately positive (ranging from 0.44 to 0.83), as well as statistically reliable as seen in Table 2, with the exception of the combining like terms skill. These correlation results show that the inverse planning algorithm approximates how teachers would rate participants’ proficiency for five of the six skills. The poor correlation between teacher and algorithm rating on the combining like terms skill is likely due to two issues. First, unlike the

Table 1
Pearson correlations between each pair of teachers. 0.5 and above indicates a moderate positive linear relationship between mean teacher rating and algorithm rating; 0.7 and above indicates a strong positive relationship

	Teacher 1 vs. 2		Teacher 1 vs. 3		Teacher 2 vs. 3	
	r	*p*	*r*	*p*	*r*	*p*
Moving terms	.61	<.001	.56	<.01	.38	<.05
Combining like terms	.43	<.05	.06	.73	.42	<.05
Isolating the variable	.64	<.001	.75	<.001	.59	<.001
Distribution	.59	<.001	.15	.44	.40	<.05
Arithmetic	.62	<.001	.56	<.01	.35	0.06
Planning	.72	<.001	.66	<.001	.56	<.01

teachers, the algorithm models combining unlike terms as a binary value: Someone either considers it possible to combine unlike terms or does not. While the inference algorithm could find a mean value between zero and one, this rarely occurs in practice because of the underlying model. Additionally, examination of the problems solved by some learners shows that when the learner takes actions that the model has no explanation for, it can in some cases misinterpret these actions as combining unlike terms. Thus, learners who actually combine unlike terms and those not well fit by the model may both receive scores of zero for this parameter. These issues likely explain why the algorithm's assessment is not positively correlated with teacher scores on this skill, nor does it add to the variance in a linear model predicting the mean teacher rating on this particular skill.

5.2.2. Predicting teacher ratings from model output

To further explore how the output of the inverse planning model was related to teacher's assessments, we used three linear regression models for each skills to determine whether the algorithm provides relevant information about a participant's performance beyond the number of problems they solved correctly (see Table 3). In this analysis, we used the "no stop" version of the algorithm, which performed better than "stop" version, likely because teachers did not view giving up as strongly related to deficiencies in planning (unlike inverse planning does). In all three models, the mean teacher rating for each skill was the dependent variable. The first model predicts the mean teacher assessments based solely on the number of problems correctly solved; the second makes the same prediction based on number correct and the algorithm's assessment for that specific skill; the third makes this prediction from number correct and all of the algorithm assessments (not just the one for the associated skill).

Adding the associated algorithm assessments as predictors of mean teacher assessments in addition to number correct provides a better approximation of teacher scores overall. As shown in Table 3, for all skills, the second model has lower AIC, and for moving terms, arithmetic, planning, and isolating the variable, the difference in AICs indicates at least some positive evidence for the model that includes the algorithm assessments being

Table 2
Pearson correlations between algorithm and the mean of each teacher's score on each of the six skills and associated p values for the "no stop" (left) and "stop" (right) versions of the algorithm. 0.5 and above indicates a moderate positive linear relationship between mean teacher rating and algorithm rating; 0.7 and above indicates a strong positive relationship

	No Stop		With Stop	
	r	*p*	*r*	*p*
Moving terms	.78	<.001	.49	<.01
Combining like terms	−.14	.46	.06	.74
Isolating the variable	.83	<.001	.68	<.001
Distribution	.55	<.01	.43	<.05
Arithmetic	.67	<.001	.71	<.001
Planning	.63	<.001	.63	<.001

a better fit. All models that include the algorithm output have a higher R^2, showing that more of the variance is explained by this model: The algorithm's assessment provides additional predictive power beyond that provided by number of problems correct. Five of the six skills have AICs that are much larger for the model that includes algorithm assessments for all six skills, as these versions increase model complexity without significantly improving fit.

Overall, these results demonstrate that the inverse planning algorithm is capturing similar information to teachers when both rate students on the same skills, and that the algorithm captures variance in teachers' scores that is not predicted by a simpler score like the number of problems answered correctly. This strengthens the evidence from Experiment 1 about the algorithm's accuracy by showing that when teachers evaluate student equation solving across multiple problems, they come to similar conclusions as those of the algorithm.

6. Selecting guidance using inverse planning

The first two experiments demonstrate that an inverse planning model is able to interpret people's equation solving and its assessments are generally consistent with assessments by teachers. One of the goals of having a model that can interpret people's patterns of problem solving is to use that information to automatically customize learners' experiences in our technology. In particular, the model's assessment could be used to select appropriate formative feedback for learners. While formative feedback to learners is an effective way to improve understanding and help create a more integrated base of knowledge (Shute, 2008), the problem of determining what type of feedback will be most effective is an area of active research. Much of the previous work on feedback in mathematics tutors has focused on progressively more informative hints (e.g., Koedinger & Aleven, 2007). More holistic information based on assessments of skills may be provided to students, such as when making a learner model "open" to the learner (Bull & Kay,

Table 3
Akaike information criteria and adjusted R^2 for three linear models. Models predict mean teacher rating based on (1) number correct, (2) number correct and assessment from algorithm on this skill, and (3) number correct and assessment from algorithm on all skills. ΔAIC is the difference between AICs for pairs of models. ΔAIC between zero and 2 indicates weak evidence for the model with lower AIC; ΔAIC between 2 and 4 indicates some positive evidence; ΔAIC between 4 and 7 indicates substantial positive evidence; ΔAIC between 7 and 10 indicates strong evidence; and ΔAIC > 10 indicates very strong evidence (Burnham & Anderson, 2004)

Skill	Model 1: Num. Correct	Model 2: Num. Correct & Algorithm	ΔAIC: Model 1 Versus Model 2	Model 3: Num. Correct & All alg. Assessments	ΔAIC: Model 1 Versus Model 2
Moving terms	AIC = 236.6 R^2 = .506	227.0 0.652	9.6	230.3 0.658	−3.3
Combining like terms	AIC = 228.4 R^2 = .718	225.3 0.753	3.1	229.6 0.750	−4.3
Isolating the variable	AIC = 225.2 R^2 = .618	224.6 0.635	0.6	228.8 0.631	−4.2
Distribution	AIC = 222.9 R^2 = .757	208.2 0.856	14.7	215.9 0.836	−7.7
Arithmetic	AIC = 203.3 R^2 = .708	202.3 0.727	1.0	201.3 0.768	1.0
Planning	AIC = 226.6 R^2 = .798	219.8 0.844	6.8	227.3 0.824	−7.5

2007), but this is not necessarily paired with feedback or interventions to remediate understanding. Research about teachers' responses to student work in educational technologies has found that teachers may customize their instruction in a variety of ways to adjust to student misunderstandings (Matuk, Linn, & Eylon, 2015). Based on this work, we were interested in how more holistic feedback that focuses on a particular skill that a student is struggling with, rather than a specific problem, might affect learning.

Given the output of the inverse planning algorithm, our goal was to "close the loop" by providing learners with a feedback activity that could help to remediate their understanding of a particular skill. To minimize differences in feedback effectiveness due to quality rather than topic, all of the feedback interventions followed the same pattern. The resulting interventions targeted five of the six skills, omitting planning. We did not create an intervention related to planning proficiency due to the fact that this is the only skill not focused on a particular type of algebra or arithmetic proficiency but instead is focused on integrating actions together. Because of this difference, it seemed likely that effective feedback about planning may need to follow a somewhat different pattern than feedback for the other five skills.

Each feedback intervention included two parts: a tutorial including information about the skills in written and video form, and opportunities for problem-solving practice. First, the interventions began with an overview screen that showed the learner two skills: The first skill was the one closest to mastery, and the second skill was the one she would receive feedback about. In both cases, she was shown her proficiency level as a colored bar and a short description of the skill. The bottom of the page told her that she would be learning more about the second skill that was shown; we refer to this skill as the *feedback skill.* On the next page, learners were shown a short embedded video about the feedback skill. Since there already exist a large number of freely available educational videos, we aimed to connect learners to a relevant resource rather than create new tutorial content. All videos were sourced from Khan Academy and were chosen because they targeted one of the five skills and were of relatively short length (2–5 min).

After the video, several stages of scaffolded practice were provided. For the four skills related to algebraic rules, the scaffolded problems were designed as follows: The first four problems highlight the core skill being practiced. For example, only the feedback focused on correctly applying the distributive property included practice on the distributive property. For these problems, the learner's steps were checked for correctness with each new step. If a mathematical error was detected, the step was highlighted and she was asked to fix it before continuing. After each problem, the learner was told the correct answer. Following these problems, eight problems were provided that still focused on the feedback skill, but checking of the learner's steps was only provided after she submitted her answer. At that point, any steps with errors were highlighted and the learner was given the opportunity to review them before continuing. These problems thus still targeted the feedback skill, but included slightly less immediate assistance than the first set of problems. For the feedback targeting arithmetic, all 12 practice problems were arithmetic computations to complete rather than algebraic equations. Finally, all feedback finished by providing 12 algebra problems that were not specialized based on the feedback skill, with the intention for people to practice in context what they had learned from the skill-specific problems. The interface for these problems was the same as when doing general problem solving on the website: People had the opportunity to enter in each individual step of the problem before continuing, and they were told whether they were correct before moving to the next problem.

7. Experiment 3: Selecting guidance

In this experiment, we explored whether it is helpful to select which activity a person should complete based on the algorithm's assessment of their skills. While it is intuitively plausible that personalized feedback based on this assessment might be more helpful than non-personalized feedback, there are several reasons to be skeptical. First, the algorithm's assessment is an approximation: There is an error due to the fact that the posterior must be approximate as well as error in the model itself. In general, the algorithm can interpret most problem solutions, but some people's behavior may be poorly fit by the model, resulting in poor accuracy for some individuals. Additionally, the algorithm does not account for learning within the period that the skills are being assessed, and depending on the person's behavior, there may be some skills about which we have very limited information. For example, as we saw in Experiment 1, the assessment for an individual may have some dimensions with relatively large confidence intervals for possible skill proficiencies due to the need for these skills occurring infrequently or a participant giving up on most problems that use these skills. A second concern about personalizing feedback is that learners who are struggling may be struggling in many skills. In that case, it may be that the personalization is unnecessary: Most students who benefit from one feedback activity would also benefit from any of the other feedback activities. Thus, we ran an experiment to test whether the feedback activities were associated with learning and to examine whether personalized feedback produced larger learning gains than feedback that was not personalized based on the algorithm's assessment.

7.1. Methods

7.1.1. Participants

A total of 200 participants in the United States were recruited from AMT and compensated $4 for session 1, $6 for session 2, and $8 for session 3. As in previous experiments, participants affirmed that they had taken an algebra course and had not completed college math classes beyond algebra.

7.1.2. Stimuli

Participants completed a multiple-choice assessment, solved algebra problems on a website, and responded to several surveys. The 12-question multiple-choice assessment was based on College Board ACCUPLACER® tests. The questions were substantially similar to the Elementary Algebra questions used in Experiment 1, but parallel versions (counterbalanced) were used for pre- and posttests.

All problem solving on the website used a similar interface to that in previous experiments. In sessions 1 and 3, learners were told whether or not they were correct immediately after submitting a problem. During the feedback in session 2, the interface behaved as described in the previous section.

The participants completed the same questions about their demographics and mathematical background as in previous experiments, and they completed a version of the usability survey from Experiment 1, augmented to add several questions about the perceived helpfulness of the feedback.

7.1.3. Procedure

Participants completed three sessions, separated by at least 1 day. In the first session, all participants solved 24 equations on the algebra website, followed by the multiple-choice questions about elementary algebra topics. The website included a short tutorial about how to use the interface, and the 24 problems were generated following the same procedure as in Experiment 2. After a participant completed all problems on the website, the assessment for that participant was computed automatically by the inverse planning algorithm, and the results were stored in the database for the participant's next session. Participants were randomly assigned to receive either version one or version two of the multiple-choice questions; these versions were identical except for changes to the exact numbers used in the problems.

In the second session, participants completed one of the feedback activities. They were randomly assigned to either *targeted* or *random* feedback. Those receiving targeted feedback completed the feedback activity for the skill which the algorithm estimated they had least proficiency; those receiving random feedback completed one of the five feedback activities selected uniformly at random.

In the third session, participants again solved 24 equations on the algebra website, followed by the multiple-choice questions about elementary algebra topics. Just as in the first session, participants all completed problems on the website that used the same templates. For the multiple-choice questions, each participant received the version of the questions that they were not assigned in the first session. Finally, participants ended the third session by completing the demographics and usability surveys.

7.2. Results

Eighty-two percent of participants completed all three parts of the experiment, with the remaining 18% either not completing a later session or, in a small number of cases, experiencing technical problems (e.g., needing to restart the computer during a session and thus losing their place in the activity). The results that follow include only the 164 participants (83 in the targeted condition and 81 in the random condition) who completed all parts of the experiment without technical difficulties.

Responses to our demographics questions suggest that participants came in with varying levels of mathematics background and that for most, significant time had passed since they had last studied algebra in school. Ninety-eight percent of participants reported what previous math classes they had taken, in college or in high school. Sixty-two percent of those who responded had taken no math classes beyond geometry (typically at a high school level); the remaining participants had taken trigonometry, pre-calculus, or calculus at a high school level. A number of participants who reported taking one of these higher-level courses in high school also reported taking a college algebra class. Thus, we would expect all participants to have prior experience with solving equations, but to be likely to have some gaps in their knowledge. This is exactly the type of person that the guidance is intended to assist.

We first examined changes in participants' performance between the first session, before getting feedback, and the final session, after getting feedback. Results from the first session confirmed that participants were on average far from ceiling on the task: They correctly answered an average of 7.2 ($SE = 0.29$) multiple-choice questions out of a total of 12, and correctly answered an average of 13.4 ($SE = 0.51$) out of the 24 algebra problems on the website. There was a small increase in the number of multiple-choice questions answered correctly in the final session. Using a repeated-measures ANOVA with factors for time of test, condition, and a random factor for participant, we found that this main effect was reliable ($F(1,162) = 15.7$, $p < .001$), but there was no interaction between condition (targeted vs. random feedback) and time of test. Given that many of the questions focused on skills that were not directly targeted by our intervention, including some quadratic equations and linear inequalities, it is not surprising that we see only a small improvement from the first to the final session. The increase in performance was somewhat larger for the algebra equations solved on the website: Participants correctly answered 23% more problems correctly, for a mean of 16.6 (*SE* = 0.49) problems correct in the final session. We again used a repeated-measures ANOVA with factors for time of test, condition, and a random factor for participant to analyze the reliability of this finding, and we found that there was a main effect for time of test ($F(1, 162) = 89.9$, $p < .001$; partial $\eta^2 = 0.36$), but no interaction between time of test and condition ($F(1, 162) = 9.46$, $p = .3$; partial $\eta^2 = 0.006$).

To better understand why there was no interaction between condition and the amount of improvement, we examined the estimated proficiency level of the skills for which feedback was given. On average, the targeted condition selected skills that had lower levels of proficiency (average proficiency level of 0.56, *SE* = 0.04, for the feedback skill in the targeted condition vs. 0.88, *SE* = 0.02, in the random condition; $t(162) = 7.03$, $p < .001$, Cohen's $d = 1.1$), indicating that in many cases, there were large differences between the least mastered skill and a random skill. However, there were a number of participants in the random condition who received feedback about a skill with which they were struggling as well as participants in the targeted condition who were close to mastery for all skills. To test whether participants who received feedback that was more appropriate for them improved more than participants who received feedback that was less appropriate for them, we divided all participants into two categories: those who received feedback about a skill that was estimated to be less than a proficiency level of 0.85 (an *unmastered* skill) and those who received feedback about a skill that was at a proficiency level ≥0.85 (a *mastered* skill). This criterion categorizes 46% of participants as receiving feedback about an unmastered skill. As shown in Fig. 6, participants who received feedback about an unmastered skill improved more than those who received feedback about a mastered skill (average improvement for those who got feedback on an unmastered skill: 4.2, *SE* = 0.52; average improvement for those who got feedback on an mastered skill: 2.2, *SE* = 0.40). A repeated-measures ANOVA with factors for whether the feedback skill was already mastered, time of test, and a random factor for the participant showed that there was a main effect of time of test as well as an interaction between time of test and whether the feedback skill was already mastered ($F(1, 162) = 9.42$, $p < 01$; partial $\eta^2 = 0.05$). To ensure that this result was not simply due to the cutoff level we chose for mastery, we checked the robustness of this result across the full range of possible mastery levels and found the same trends for mastery levels both somewhat higher and somewhat lower than this cutoff; see Appendix G for details. While these results must be interpreted with some caution, as participants were not randomly assigned to the two categories, they suggest that receiving feedback that the algorithm indicates is more appropriate can result in greater improvements in performance.

Based on the fact that proficiency level influenced the effectiveness of the feedback, we examined the distribution of proficiencies for individual participants. We were interested in whether participants tended to have all skills at a similar level or whether they usually had some skills that were mastered and some that were unmastered. As shown in

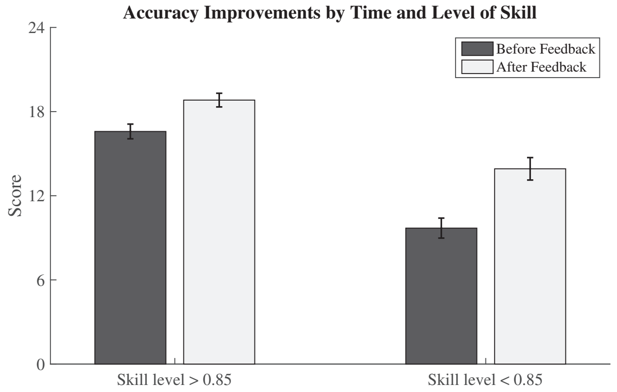


Fig. 6. Improvement from first to last session in accuracy on website problems, categorizing participants based on prior level of proficiency in feedback skill. Participants who received feedback about an unmastered skill improved more from the first to the final session than those who received feedback about a mastered skill.

Fig. 7, 35% of participants were at mastery for all skills, where mastery is defined as proficiency of at least 0.85, and 14% of participants were not at mastery for any skills. The remaining 51% of participants ($N = 81$) who had some mastered skills and some unmastered skills are arguably those that might most benefit from targeted rather than random feedback. A repeated-measures ANOVA with factors for time of test, condition, and a random factor for participant shows that there is a significant interaction between time of test and condition when restricting the data to these participants:[6] As shown in Fig. 8, those who completed targeted feedback improved almost twice as much those who completed random feedback (average improvement 5.3, $SE = 0.11$, vs. 3.0, $SE = 0.10$; $F(1, 82) = 5.64$, $p < .05$; partial $\eta^2 = 0.06$[7]). This suggests that inverse planning can provide a benefit for these participants: It allows us to determine what skill(s) will be appropriate targets for feedback.

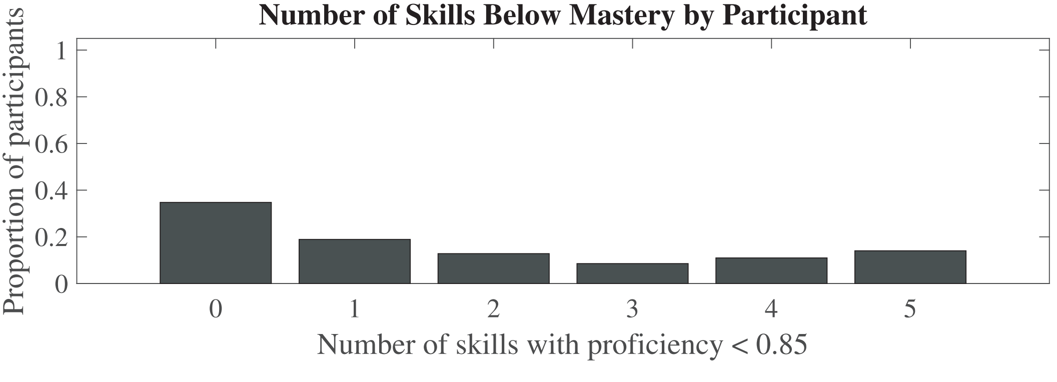


Fig. 7. Count of the number of unmastered skills by participant.

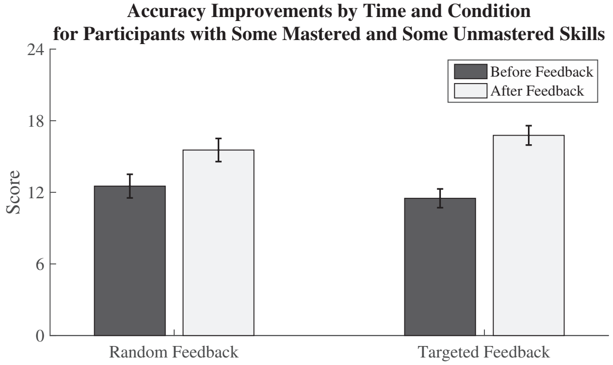


Fig. 8. Improvement from first to last session in accuracy on website problems, restricted to participants with some unmastered and some mastered skills. Participants show reliable improvement, and participants who received targeted feedback tended to improve more than those who received random feedback.

8. Using production rule models in inverse planning

To use inverse planning to model people's equation solving skills, we focused on six dimensions of proficiency. However, much more complicated models of equation solving are possible, and these dimensions certainly do not capture all of the possible systematic differences in people's skills and beliefs about allowed equation transformations. To demonstrate that the inverse planning framework can scale to existing alternative models of mathematics skills, we now show how a production rule model can be translated into the inverse planning framework to recognize individual students' problem-solving patterns.

Production rule models are common in many educational domains, and we specifically focus on a production rule model of fraction arithmetic skills with over twice as many mal-rules and strategies as in our equation solving model (Braithwaite, Pyke, & Siegler, 2017). We first describe our application of inverse planning to fraction arithmetic, illustrating how to translate the production rule model in Braithwaite et al. (2017) into a hypothesis space characterizing differences among individuals and an MDP for fraction arithmetic that is parameterized by a hypothesis. We then present the results of applying inverse planning to three datasets of people's fraction arithmetic solutions.

8.1. Applying inverse planning to fraction arithmetic skills

Braithwaite et al. (2017) developed a set of production rules for how students solve fraction arithmetic problems, including rules focused on strategy choices and rules focused on execution. Their goal was to create a model that replicated a large set of existing empirical observations about students' proficiency with fraction arithmetic, including their struggles to identify the right strategies to use for each problem and variable use of strategies both between students and for individual students. Their model focuses on arithmetic problems involving two operands, both fractions. Strategy rules are invoked at the beginning of problem solving to decide what series of production rules should be followed, and each rule has a particular context in which it can be invoked. For instance, the *Op-deleted Add/Sub with equal denominators* rule can be invoked when no other rule has yet been chosen and the denominators of the operands are equal, and if it is invoked, subgoals are created to set the denominator of the answer equal to the common denominator and the numerator of the answer equal to the result of performing the operation in the problem on the numbers of the operand. For instance, if the problem was $\frac{6}{4} \div \frac{3}{4}$ and this rule was invoked and no execution mal-rules were used, then the final answer would be $\frac{2}{4}$ (which might be simplified, depending on later choices).

While the work in Braithwaite et al. (2017) focused on developing a model that would *learn* to solve fraction arithmetic problems by learning weights on which production rules to use, our focus is on identifying tendencies toward particular mal-rules for individuals. Thus, we use their production rule model of the possible rules, but we do not incorporate their model for learning which rules to use. Instead, we use the production rules to create a hypothesis space and then make inferences about individuals based on their solutions to problems.

To use the production rules of Braithwaite et al. (2017) in inverse planning, we need to define an MDP based on these production rules and define how individuals' tendencies toward using incorrect production rules (mal-rules) is incorporated. First, we define the states, actions, transition function, and reward function for the fraction arithmetic MDP. In the model in Braithwaite et al. (2017), fraction arithmetic is formulated as the execution of production rules in particular contexts based on the current goals, with each production rule potentially influencing the current goals. Translating this into an MDP framework, we define each state by the current goals, the problem to solve, and the (partially completed) answer. The start state always has the set of goals equal only to *[find the answer]* and the answer as a blank fraction, awaiting filling in the numerator and the denominator.

The actions in the MDP are firing a particular production rule. Braithwaite et al. (2017) define the conditions under which a particular production rule can be fired. These conditions can include features of the operands (e.g., if their denominators are equal), features about progress in the problem (e.g., whether the numerator of the answer has been filled in), and which goals are currently defined in the state (see Braithwaite et al. (2017) for tables of the 36 production rules in their model and the contexts in which they can be used). For each state, we include actions for the production rules that are allowed in that state.

Like when modeling equation solving skills, the transition model is dependent on the particular hypothesis $\theta \in \Theta$ characterizing the individual. In this case, that hypothesis corresponds to their tendency toward particular types of fraction arithmetic mal-rules. The hypothesis space Θ should represent the space of possible tendencies toward different mal-rules, including both strategy and execution mal-rules. Thus, we use one parameter for each strategy rule as well as each execution mal-rule. Each parameter is represented as a different dimension of a hypothesis θ, and how these parameters contribute to the likelihood of observing a particular solution for a problem is dependent on whether the rule is a *strategy* rule or an *execution* rule.

Strategy rules: The strategy rules are identical to those in Braithwaite et al. (2017). Four rules reflect normative problem-solving strategies for each type of problem. We refer to these rules as *Correct Add/Sub with equal denominators*, *Correct Add/Sub with unequal denominators*, *Correct Mult*, and *Correct Div*. Four strategy mal-rules involve applying a correct strategy for one type of problem but ignoring whether the current problem is in fact of that problem type. For instance, *Op-deleted Add/Sub with equal denominators* can be executed for any problem where the operands have equal denominators, and it creates goals of setting the answer numerator equal to the result of applying the current operator to the numerators of the operands and setting the answer denominator equal to the denominator of the operands. If this strategy were applied to the problem 4/5 × 3/5, then if all other operations were carried out correctly, the result would be 12/5. There is an *Op-deleted* version of each of the correct actions. Finally, Braithwaite et al. (2017) include a *Cross-multiply* strategy rule, applicable only for multiplication problems. This rule sets the numerator of the answer to the product of the first numerator and the second denominator, and the denominator to the product of the second numerator and the first denominator.

Each strategy rule is associated with a real-valued parameter, where higher values of that parameter indicate an individual is more likely to use that strategy rule on a problem where the rule is allowed. Rather than including one action for every strategy rule, we have one action that can be taken when the only goal is *find the answer*, and executing each possible strategy rule is a possible outcome of that action. When a particular rule is executed, the state is then changed based on how that production rule is defined in Braithwaite et al. (2017).

The probability of each strategy rule is governed by a softmax over the tendencies for all possible strategy rules for the current problem. Specifically, if strategy rules $r_1, \ldots,$ r_n are applicable to the current problem and these have corresponding tendencies $\theta_{r_1}, \ldots,$ θ_{r_n}, the probability of applying one of these strategy rules r_i is:

$$p(s_i|\text{current}\,\text{problem}) = \frac{\exp(\theta_{t_i})}{\sum_{j=1}^{n}\theta_{t_i}}. \tag{3}$$

Encoding the strategy choices in this way is consistent with the phenomenon of individuals showing variability in strategy (Braithwaite et al., 2017). For any particular problem, the probability a student will execute the correct strategy is thus a function of multiple dimensions of the hypothesis space: the dimension representing their tendencies toward the correct strategy, and the dimensions representing their tendency toward the incorrect strategies.

Execution rules: Just as for strategy rules, we follow Braithwaite et al. (2017) for the execution rules. Here, we create a parameter for each of the six execution mal-rules from Braithwaite et al. (2017): (a) failing to multiply the numerators as well as the denominators when converting to a common denominator; (b) failing to invert the second operand when executing a division strategy (either *CorrectDiv* or *Op-deleted Div*); (c) inverting the first operand instead of the second operand when executing a division strategy; (d) always subtracting the smaller number from the larger when executing whole number subtraction, regardless of the order of the operands; (e) always dividing by the smaller number executing whole number division, regardless of the order of the operands; and (f) dropping the remainder when executing whole number division. Because the two mal-rules about inverting can be applied in exactly the same situations, we constrain the parameters for these rules so that they cannot sum to a value >1. All parameter values are the probability of following that mal-rule in a situation where it is relevant to the chosen action. Just as for strategy rules, once an execution rule (correct or incorrect) is chosen, the state changes are executed according to the production rule.

9. Analyzing fraction arithmetic performance

To show that the inverse planning model can interpret people's fraction arithmetic solutions and determine whether this model is effective at capturing their tendencies toward different types of errors, we performed two analyses. First, we applied the model to the two datasets from Braithwaite et al. (2017), where solutions have been hand-coded based on both what the student wrote and how they described what they were doing as they solved the problems. However, students only solved a small number of problems in these datasets—eight in one and sixteen in the other—so there is limited data from which to infer the values for the 15 parameters that characterize each individual student. Additionally, some mal-rules were unlikely to be demonstrated in the problems in the datasets. For example, no problems had a correct solution in which a larger number was subtracted from a smaller one, and thus there were no natural points to detect if a child was using the mal-rule of always subtracting smaller numbers from larger, regardless of problem order. Thus, we included only a restricted version of the hypothesis space for these first analyses, and in our second analysis, we collected new data in which people on AMT solved a wider range of problems and made inferences over the full hypothesis space.

In the existing data, only students' final answers were recorded, not intermediate work on the problems. Additionally, the structure of the production rule system meant that some state transitions were not possible to observe directly, and most states had some information that was unobserved. For instance, all states included *goals*, but the model does not suppose that the student expresses those goals in any explicit way. Thus, to apply the inverse planning model to these data, we assume that only the problem and the answer (i.e., the first and last states) are observed, and we marginalize over the omitted states and actions.

9.1. Analyzing Braithwaite et al. (2017) data

We first applied the inverse planning model to the two datasets in Braithwaite et al. (2017). The first (STS2011; originally reported in Siegler, Thompson, and Schneider [2011]) included solutions from 48 students, each of whom solved eight problems. The second (SP2013; originally reported by Siegler and Pyke [2013]) included solutions from 120 students, each of whom solved 16 problems. In both datasets, half of students were in sixth grade and half were in eighth grade. In addition, Braithwaite et al. (2017) provided us with their coding of the strategy taken by each student on each problem: a strategy appropriate for an addition or subtraction problem, a strategy appropriate for a multiplication problem, a strategy appropriate for a division problem, or some other strategy. Strategies were coded based on students' verbal reports, made immediately after solving each problem.

We performed three types of comparisons: The first compares the strategy parameters to the hand-coded strategies, whereas the latter two compare the strategy and execution parameters to students' problem-solving accuracy.

Comparison to coded strategies: We expect the number of times a student used each of the non-"other" strategies on different types of problems to be related to the model strategy parameters.

To compare the number of times each strategy was actually used with the parameter estimates from the inverse planning model, we computed Pearson's correlations between the count of how often each student used a particular strategy and the average probability placed on that strategy by the inverse planning model. We use model probabilities rather than raw parameter values because the *strategy* parameters are combined using a softmax function: The numeric value of each individual strategy parameter is meaningful only when considered in conjunction with the values of the other strategy parameters. Since the available strategies differ by problem type, the model probabilities for each strategy also differ by problem type. We compute correlations between each strategy parameter and two categories of problem types: those where the strategy is correct (e.g., multiplication problems for the multiplication strategy) and those where the strategy is incorrect. For those where the strategy is incorrect, we average the model probabilities for each of the problem types. Additionally, for addition/subtraction problems, there are two strategy parameters: one for problems where the operands have equal denominators and one where they have unequal denominators. We averaged across the probabilities resulting from these parameters; results are substantially the same if types of problems are instead further divided into equal and unequal denominators.

As shown in Table 4, for almost all strategies, there was a reliable, positive correlation in both datasets between the model's probability on that strategy and the number of problems on which a student used that strategy. The exception is for the division strategy on non-division problems in the STS2011 dataset. This is likely because there were very few non-division problems where students used a division strategy: In the STS2011 dataset, only three of the 48 students made this error.

Comparing accuracy and strategy *parameters*: We next explored how the inverse planning model's inferences about strategies was related to students' accuracy on different types of problems. Students who use incorrect strategies more often should be less accurate (and vice versa). As in the analyses above, we computed the model probability on each type of problem, rather than the raw parameter value. As shown in Table 5, higher probabilities on incorrect strategies for a particular type of problem tend to be associated

Table 4
Correlations between the number of times students used each strategy, according to coding of verbal reports, and the inverse planning model's probability of each strategy

Strategy	Problem Type	ST2013	SP2011
Addition/subtraction strategy	Multiplication and division	$r(118) = .880, p < .0001$	$r(46) = .705, p < .0001$
Addition/subtraction strategy	Addition and subtraction	$r(118) = .875, p < .0001$	$r(46) = .818, p < .0001$
Division strategy	Non-division	$r(118) = .339, p < .0005$	$r(46) = .0300$, p = .84
Division strategy	Division	$r(118) = .748, p < .0001$	$r(46) = .470, p < .005$
Multiplication strategy	Non-multiplication	$r(118) = .651, p < .0001$	$r(46) = .356, p = .013$
Multiplication strategy	Multiplication	$r(118) = .818, p < .0001$	$r(46) = .433, p < .005$

with lower accuracy on that type of problem, and conversely, higher probabilities on correct strategies tend to be associated with higher accuracy. Only two categories of correlations are not statistically significant: those for the division strategy on multiplication problems, and those for the multiplication strategy on division problems. The lack of correlation is partially explained by the small number of students who used these strategies on the associated problem type.

Execution parameters: Unlike the strategy parameters, the *execution* parameters are not all intrinsically linked to one another. Thus, we can evaluate the model's performance for each of them separately, with the exception of the parameters for failing to

Table 5
Correlations between accuracy on particular types of problems and probabilities of strategies on those problems. As above, we average across equal and unequal denominator probabilities for addition/subtraction problems; results are substantially the same if types of problems are instead further divided into equal and unequal denominators

Problem Type	Strategy Parameter (s)	Expected Dir.	SP2013	STS2011
Addition/ subtraction	OpDeletedMult	Negative	$r(118) = -0.639, p < .0001$	$r(46) = -.431, p < .005$
Addition/ subtraction	OpDeletedAddSub, CorrectAddSub	Positive	$r(118) = .865, p < .0001$	$r(46) = .799, p < .0001$
Addition/ subtraction	OpDeletedDiv	Negative	$r(118) = -.562, p < .0001$	$r(46) = -.536, p < .0001$
Division	OpDeletedMult	Negative	$r(118) = -.0523, p = .57$	$r(46) = -.212, p = .15$
Division	OpDeletedAddSub	Negative	$r(118) = -.501, p < .0001$	$r(46) = -.319, p = .027$
Division	OpDeletedDiv, CorrectDiv	Positive	$r(118) = .492, p < .0001$	$r(46) = .446, p < .005$
Multiplication	OpDeletedDiv	Negative	$r(118) = -.0798, p = .39$	$r(46) = -.116, p = .43$
Multiplication	CrossMultiply	Negative	$r(118) = -.252, p < .005$	$r(46) = -.404, p < .005$
Multiplication	OpDeletedMult, CorrectMult	Positive	$r(118) = .838, p < .0001$	$r(46) = .550, p < .0001$
Multiplication	OpDeletedAddSub	Negative	$r(118) = -0.71366, p < .0001$	$r(46) = -.30018, p = .038$

invert or inverting the wrong operand when executing a division strategy, which we examine together. One way of assessing the model's performance in inferring values for these parameters would be the size of the correlation between each execution mal-rule parameter and the type of problem where the option of using this mal-rule is most likely to occur. However, while these correlations are present in our data, they ignore the fact that accuracy across problem types is often correlated, so the values of the mal-rule parameters are often also highly correlated with the accuracy for other types of problems. For instance, for the execution mal-rule involving subtracting smaller numbers from larger, regardless of the actual order of the problem, there is as expected a strong negative correlation between the tendency toward this mal-rule and accuracy on subtraction problems in the SP2013 data ($r(118) = -.59$, $p < .0001$), but this tendency is also strongly negatively correlated with accuracy on non-subtraction problems ($r(118) = -.34$, $p < 0.0005$).

To assess whether larger tendencies toward particular mal-rules are *specifically* associated with poorer performance on particular types of problems, we use logistic regression to predict accuracy on each individual problem. We perform one logistic regression for each mal-rule (except for combining the two mal-rules related to inversion), and each logistic regression includes factors for the mal-rule value, whether the problem is of the most impacted type, an interaction between these two factors, and a random factor for participant. If a mal-rule is in fact associated with performance on a specific problem type, then the interaction term will be reliably different from zero.

Overall, as shown in Table 6, higher probabilities of using a particular mal-rule tend to be associated with lower accuracy on the problem type most impacted by the mal-rule. However, not all interaction terms are reliably different from zero, and in one case (*divide larger by smaller*), the term has the opposite of the predicted sign. The discrepancies are particularly acute for the STS2011 dataset, where students solved only eight problems. It is thus unlikely that there is sufficient data to obtain precise estimates of the probabilities with which students use particular execution mal-rules, given that the mal-rules may arise only in specific types of problems and may sometimes be relevant only when an incorrect strategy is used.

9.2. Experiment 4: Analysis of larger fraction arithmetic dataset

Because the datasets from Siegler et al. (2011) and Siegler and Pyke (2013) contained a limited number and range of problems, we followed our analyses above by collecting a larger dataset and then applying the inverse planning model to the collected data.

9.2.1. Methods

9.2.1.1. Participants: A total of 96 participants in the United States were recruited from AMT and compensated $3.50.

9.2.1.2. Stimuli: Participants completed 80 arithmetic problems. Following the previous datasets, all problems had two operands, both fractions. There were 20 problems for each

Table 6
For three of the five kinds of execution mal-rules, higher probabilities of using a mal-rule are associated with lower accuracy on the problem type most impacted by that mal-rule in at least one of the SP2013 or STS2011 datasets. Reported values are for the interaction between problem type and mal-rule probability in a logistic regression predicting accuracy on each problem

		Coefficient for Interaction Term	
Execution Mal-rule	Most Impacted Problem Type	SP2013	STS2011
Multiply only denom. (not numerators) when converting denom	Add./sub. With unequal denom.	Value = −8.47, *t* (1916) = 7.47, *p* < .0001	Value = −25.8, *t*(380) = 4.26, *p* < .0001
Subtract smaller from larger (ignoring ordering)	Subtraction	Value = −3.67, *t* (1916) = 5.27, *p* < .0001	Value = −0.00862, *t* (380) = 0.00230, *p* = .999
Divide larger by smaller (ignoring ordering)	Division	Value = −0.756, *t* (1916) = 0.979, *p* = .33	Value = 2.35, *t*(380) = 0.699, *p* = .49
Drop remainder when dividing whole numbers	Division	Value = −0.346, *t* (1916) = 0.457, *p* = .65	Value = −3.34, *t*(380) = 0.854, *p* = .39
Inversion mal-rules	Division	Value = −6.61, *t* (1916) = 7.36, *p* < .0001	Value = −21.2, *t*(380) = 3.51, *p* = .0005

operation (addition, subtraction, multiplication, and division), and within each operation, there were 10 problems in which the operands had equal denominators and 10 in which they had unequal denominators. Rather than using templates as in Experiments 1–3, all participants completed the same arithmetic problems, and the problem set was designed so that for many problems, there were multiple ways to get the same incorrect answer. This was intended to make the dataset as ambiguous as possible for the inverse planning model, and thus provide the best test of whether it could make sense of data where there were multiple possibilities for the cause of an error.

9.2.1.3. Procedure: Participants completed the problem in a single session online and entered their responses in the interface of the algebra website, with fraction arithmetic problems rather than equations. The order of the problem set was randomized for each participant.

9.2.2. Results

Because we do not have verbal reports of participants' strategies, we focus on how accuracy on different problem types is associated with the mean parameter values inferred by the model.

Strategy parameters: As above, we examine the correlation between the inverse planning model's average inferred probability of using each strategy and accuracy on different problem types. As shown in Table 7, higher probability on a strategy that is correct for a problem type is reliably associated with higher accuracy on that problem type, and higher probability on an incorrect strategy type is reliably associated with lower accuracy on that problem type. These correlations are consistent with the results from the analyses of prior data but also find that the correlations for the division strategies are reliable; this likely occurs because of the larger dataset: Each participant in this study solves 20 division problems, rather than only two or four division problems.

Execution parameters: Table 8 shows that for the larger dataset, we also find that with the exception of the two division mal-rules that can only be used when using an incorrect strategy, higher probabilities of using each execution mal-rule are associated with lower accuracy specifically on the most impacted problem type. For the two division mal-rules that only come up in fraction arithmetic when the student is using an incorrect strategy, the interaction term in the logistic regression is in the expected direction and is marginally significant.

9.3. Discussion of fraction arithmetic application

By applying inverse planning to fraction arithmetic, we have shown how an existing production rule system for mathematical skills can be translated into the inverse planning framework. Applying the model to the two existing datasets demonstrates that while the model is complex, its inferences are consistent with manual coding of students' strategies, providing an automated way to capture students' strategies (including the amount of variability in those strategies). The analysis of the larger dataset replicated the findings from the smaller datasets and also demonstrated that with sufficient data per student, fine-grained inferences can be made about a large number of parameters. While these results are limited in that they do not fully take advantage of the inverse planning model's ability to interpret step-by-step solutions to individual problems, they provide evidence for how inverse planning can be used with larger, more complex hypothesis spaces and make use of existing modeling approaches.

Table 7
Correlations between accuracy on particular types of problems and probabilities of strategies on those problems for the larger fraction arithmetic dataset

Problem Type	Strategy Parameter(s)	Expected Dir. of Corr.	Correlation
Addition/subtraction	OpDeletedMult	Negative	$r(94) = -.702$, $p < .0001$
Addition/subtraction	OpDeletedAddSub, CorrectAddSub	Positive	$r(94) = .832$, $p < .0001$
Addition/subtraction	OpDeletedDiv	Negative	$r(94) = -.655$, $p < .0001$
Division	OpDeletedMult	Negative	$r(94) = -.417$, $p < .0001$
Division	OpDeletedAddSub	Negative	$r(94) = -.480$, $p < .0001$
Division	OpDeletedDiv, CorrectDiv	Positive	$r(94) = .580$, $p < .0001$
Multiplication	OpDeletedDiv	Negative	$r(94) = -.286$, $p < .005$
Multiplication	CrossMultiply	Negative	$r(94) = -.449$, $p < .0001$
Multiplication	OpDeletedMult, CorrectMult	Positive	$r(94) = .624$, $p < .0001$
Multiplication	OpDeletedAddSub	Negative	$r(94) = -.437$, $p < .0001$

Table 8
For all execution mal-rules, higher probabilities of using a mal-rule are associated with lower accuracy on the problem type most impacted by that mal-rules; for three of the five kinds of execution mal-rules, this association is statistically significant, while for the other two, the association is marginally significant. Reported values are for the interaction between problem type and mal-rule probability in a logistic regression predicting accuracy on each problem

Execution Mal-rule	Most Impacted Problem Type	Coefficient for Interaction Term
Multiply only denominators (not numerators) when converting to a common denominator	Add./sub. With unequal denom.	Value = −2.08, $t(7{,}594) = 5.37$, $p < .0001$
Subtract smaller from larger (ignoring ordering)	Subtraction	Value = −1.33, $t(7{,}594) = 6.76$, $p < .0001$
Divide larger by smaller (ignoring ordering)	Division	Value = −0.152, $t(7{,}594) = 1.62$, $p = .1051$
Drop remainder when dividing one whole number by another	Division	Value = −0.150, $t(7{,}594) = 1.60$, $p = .1090$
Inversion mal-rules	Division	Value = −2.81, $t(7{,}594) = 12.3$, $p < .0001$

10. General discussion

The steps that learners take to solve problems can offer insight into their understanding. We used the Bayesian inverse planning framework to create a generative model for how a learner's understanding is related to the way she solves equations. We then explored both the accuracy of this model's assessments, finding that the assessments were consistent with teachers' judgments; this suggests that the model's inferences could be helpful in giving teachers more information about their students without requiring them to painstakingly examine each student's individual work. We then demonstrated that the model's assessments could be used to individualize learners' experiences. Finally, to show that inverse planning can be scaled to a larger space of misunderstandings, we adapted an existing production rule system for fraction arithmetic into the inverse planning framework, allowing detection of individual tendencies toward erroneous fraction arithmetic strategies. In the remainder of this paper, we discuss both limitations of the current results and future directions for extending this work.

10.1. Limitations

There are several limitations to the current results. First, the learners in our experiments were recruited on AMT. AMT participants may have different motivations than typical algebra students. They could be less motivated and are less likely to be currently engaged with mathematics. However, adult learners are a common population that educational technologies are particularly suited to reach. For example, algebra is frequently taken at community colleges and needed for many people attempting to switch careers or move to higher-skilled jobs. The varied backgrounds and ages of our participants may be similar to the types of variations that one would expect in such adult learners.

Additionally, while some participants may have been disengaged, time on task was generally consistent with participants engaging in significant effort, and comments from several participants indicated they wanted to learn more algebra or appreciated practicing old skills.

A second limitation is the accuracy of the underlying learner models that we used for equation solving and fraction arithmetic. As we note in developing the equation solving model, it makes a number of simplifying assumptions: For instance, it models some but not all mal-rules that have been documented in the literature and ignores the impact of the exact numbers on arithmetic error rate. While the fraction arithmetic model from Braithwaite et al. (2017) has been tested in a wider variety of circumstances, it also makes simplifying assumptions, including ignoring basic arithmetic errors. These inaccuracies in the models may lead to incorrect conclusions about some people's misunderstandings; we discuss this issue and possible ways to improve the models below. For the purposes of this paper, our goal was to more fully develop the inverse planning framework for the application to real-world educational domains, going beyond the mainly laboratory-based experiments where it had previously been used for inferring misunderstandings (Rafferty et al., 2015). Crucially, our final application of inverse planning demonstrated how to take a general production rule model and use it as the student model within inverse planning. This allows for testing alternative models of problem solving or of alternative misunderstandings.

A third limitation is that in Experiment 1, we focused on two extremes in a range of possible interfaces: One interface allowed participants to choose any next step they liked, whereas the other constrained both mathematical accuracy and choice of strategy. Some tutoring systems are certainly less constrained than our *corrected* interface, requiring only that steps be mathematically correct but not that they be in a normative order. Our goal in this experiment was primarily to establish whether allowing freer choices impacted the inverse planning algorithm's ability to understand learners' solutions, and to add to the existing body of literature suggesting that there are at least some cases where less constrained interfaces may be appealing. By significantly constraining the choice of actions in the *corrected* interface, we create minimal ambiguity for inverse planning, providing a best-case scenario for the algorithm to which we can compare the interface that allows more choices, and the fact that we permitted slightly suboptimal actions means participants still had some action choice.

In Experiment 3, we were focused on whether personalized feedback using the algorithm leads to greater learning gains. This experiment does not separate whether the content of a feedback intervention is helpful from whether the targeting of that feedback is accurate. We intend to further evaluate these two components to better understand what the maximum benefit of this type of feedback would be if targeting was perfectly accurate, but any evaluation of the overall effectiveness of the knowledge diagnosis-feedback loop must acknowledge that inaccuracies in the knowledge diagnosis (assessment) may lead to the personalization being less effective. By establishing that personalization can be helpful, it suggests that the targeting is accurate enough that further development of feedback activities may lead to even greater gains.

10.2. Model parameters and fit

The Bayesian approach that we have taken both closely models students' action planning and infers posterior distributions over the parameters, many of which are real-valued. There are significant advantages to this type of approach: It allows making detailed inferences about specific misunderstandings and tendencies toward strategies, and as in other Bayesian approaches, the output clearly indicates how much the evidence supports particular parameter values. However, this approach can require significant data to acquire confident estimates. This indicates that caution must be used when making comparisons between students or between the relative value of different parameters when only means of the posterior distribution are used. While small differences in parameter values may not be detectable given the number of problems likely to be solved by real students, these differences are also unlikely to be significant for informing interventions.

Using a detailed model of action planning requires strong theoretical assumptions about what actions students take. In some cases, these assumptions make it impossible to interpret particular transformations, as they do not correspond to any of our actions or errors, and in other cases, they may lead to incorrect inferences, as a transformation may be interpreted as having many errors rather than correctly recognized as not corresponding to any interpretable action. There are several possible solutions to these issues. First, transformations that cannot be interpreted could be examined to identify common patterns. In principle, this process could be automated by searching through a simple grammar of possible operations defined on individual terms (e.g., dividing a term by a coefficient) to identify what could account for each transformation; for instance, this could uncover errors like dividing only the first term on a side by a coefficient, rather than all terms. Common sequences of operations could then be added as actions or error types to the inverse planning algorithm. Approaches like that in MacLellan, Harpstead, Patel, and Koedinger (2016) could also be used to identify possible production rules.

Addressing erroneous interpretations of transformations based on existing actions is more difficult. There will always be ambiguity in terms of whether a learner truly intended to take a particular action and made an error, or whether a different action, perhaps unknown to the algorithm, was intended. One sign of poor fit can be an extremely low planning parameter in cases where a learner does not submit unsolved problems; while this could accurately reflect a student's abilities, it is likely to occur more frequently for students who take actions that the algorithm does not model. There are several additional ways of identifying poor fit that we intend to explore. We currently allow transformations that are mathematically correct (in that the same value for the unknown is valid for both steps) to be attributed to actions with arithmetic or other errors. While in principle this is possible, as two arithmetic errors might cancel each other out, it may be more likely that this is an erroneous action interpretation. Not allowing the algorithm to consider erroneous versions of actions in cases of mathematically correct transformations would lead to poorer coverage but potentially more accurate inferences. Poor fit might also be identified by the clustering of a student's arithmetic errors across transformations. If a student typically has many arithmetic errors in a transformation or very few errors, this could indicate that the transformations with many errors are in fact actions that the algorithm cannot interpret.

10.3. Future directions

In addition to improving the model based on data, there are a number of other future directions for this work. First, the work here lays a foundation for how to use inverse planning to infer misunderstandings related to equation solving and fraction arithmetic, but the same idea could be applied to other parts of mathematics. Because the framework relies on interpreting sequencing of student actions and linking them to misunderstandings, it could also be applied in game-like contexts, allowing learners' behaviors on a wide range of different types of problems to all contribute to the same estimate. Inverse planning has been applied to games in some previous work (Rafferty et al., 2014). There are also a number of areas to explore about how best to develop feedback based on the type of output given by the inverse planning model. The feedback here was relatively simple, but how to best design feedback interventions is an area of active research (e.g., Shute, 2008). The inverse planning algorithm also has the potential to provide additional data for feedback interventions. For example, it can extract examples of a learner making a particular type of error. This could be used to ask learners to reflect on what went wrong in a problem or to distinguish among different steps.

Future work will also explore more complicated strategy models. In this work, we assumed all learners were attempting to solve problems using as few steps as possible, and that suboptimality in planning should be modeled with a single parameter scaling between uniform random choices and optimal choices. However, there may be other factors at play in learners' strategies. Cognitive load may be a determining factor in what steps to take, as taking a step like "multiply by 2" when solving the equation $\frac{1}{2}x + 2 = \frac{5}{2}$ may add an extra step above the minimal number but make mental mathematics easier to execute. This could be reflected in the inverse planning model by varying the reward function, and a parameter could be added to the hypotheses to allow such factors to be inferred for individual learners. This approach could also be effective for learning other types of variations in reward functions across learners that lead to suboptimal strategies.

For equation solving specifically, we have made Emmy's Workshop[8] freely available. Emmy's Workshop is a newer version of the website used in these studies. This both potentially provides benefits for algebra learners and opportunities for research on longer-term interventions than that conducted in Experiment 3. Repeated interactions over time will allow us to investigate how student learning can be captured in a Bayesian inverse planning model: Repeatedly calculating the assessment over different windows of time offers one possible way to see evidence of student learning, but more broadly, it is possible to combine inverse planning with a computational model that incorporates learning (see Rafferty et al., 2014, for one example). Making the website freely available will also allow us to investigate its effectiveness for a broader population of learners, including ones who may be more intrinsically motivated than our experimental participants, and to determine its feasibility for use in classrooms. The teachers who participated in

Experiment 2 were excited about the possibility of being able to use the website with their own students, as they liked the idea of students completing the problems as homework and receiving personalized feedback that could reteach skills on which the students were struggling. Additionally, these teachers were interested in getting the algorithm's assessment of each student to gain a clearer picture of students' skills while not requiring them to actually grade the homework. The work in this paper to develop the system and investigate the characteristics of the algorithm's assessments provides a foundation for future, more applied work to explore the usefulness of the system in practice.

10.4. Conclusion

We developed a Bayesian inverse planning model for equation solving, extending this model to cover the technical challenges of being unable to directly observe people's actions and of there being an infinite number of possible contexts (equations) in which a person may be acting. This model builds on ideas from inverse reinforcement learning and assumes people act in a noisily optimal manner: They attempt to solve the equations but may make systematic mistakes either in individual equation solving steps or in their step choices based on misunderstandings. We have shown that this model can be used to interpret the vast majority of equation solving steps taken by participants in our experiments, and that the model's assessments are consistent with assessments made by math teachers, while taking much less time. We have also demonstrated that the model can be used to provide adaptive guidance for learners, providing an applied measure for the value of this algorithm in a larger system, and shown that existing production rule systems for other mathematical domains can be adapted for use in inverse planning. While additional testing with a broader range of learners is needed to fully establish the validity of the model's assessments, the current work shows the potential for integrating inverse planning into educational technologies, and it also provides an example of how this framework can be adapted to make use of existing models of student misunderstandings.

Acknowledgments

We would like to thank research assistants Jonathan Brodie, Sam Vinitsky, Louis Ye, and Shiyue Zhang for their programming contributions to the algebra website. This work was funded by NSF grant number DRL-1420732 to Thomas L. Griffiths and Anna N. Rafferty.

Notes

1. http://www.khanacademy.org/
2. While we have taken a binary approach to whether actions that combine unlike terms are included as possibilities, alternative models for how people consider the

action space are possible. For instance, a mixture model that assumes people may remember not to combine unlike terms only on some problems is possible, and such a model might admit more efficient inference using MCMC algorithms other than the Metropolis-Hastings algorithm that we use in this paper. Such a model would pose no fundamental challenges to the overall inverse planning framework.

3. The current version of the website is available at emmysworkshop.com.
4. For all models, sequences of up to three actions are permitted between steps.
5. Planning was rated on a scale from 1 to 7.
6. Forty-eight percent of these participants were in the targeted condition and 52% were in the random condition.
7. To check robustness, we also ran this test with mastery level set at 0.9. This finds the effect to be marginally significant (average improvement 4.2 vs. 2.7; F(l, 103) = 3.33, p = .07; partial $\eta^2 = 0.03$).
8. https://www.emmysworkshop.com

References

Anthony, L. (2008). *Developing handwriting-based intelligent tutors to enhance mathematics learning*. Unpublished doctoral dissertation. Carnegie Mellon University.

Anthony, L., Yang, J., & Koedinger, K. R. (2005). Evaluation of multimodal input for entering mathematical equations on the computer. In C. Gale (Ed.), *CHI'05 extended abstracts on human factors in computing systems* (pp. 1184–1187). New York: Association for Computing Machinery.

Baker, C. L., Saxe, R. R., & Tenenbaum, J. B. (2009). Action understanding as inverse planning. *Cognition*, *113*(3), 329–349.

Bellman, R. E. (1957). *Dynamic programming*. Princeton, NJ: Princeton University Press.

Braithwaite, D. W., Pyke, A. A., & Siegler, R. S. (2017). A computational model of fraction arithmetic. *Psychological Review*, *124*(5), 603.

Brown, J., & VanLehn, K. (1980). Repair theory: A generative theory of bugs in procedural skills. *Cognitive Science*, *4*(4), 379–426.

Bull, S., & Kay, J. (2007). Student models that invite the learner. In: The SMILI open learner modelling framework. *International Journal of Artificial Intelligence in Education*, *17*(2), 89–120.

Burnham, K. P., & Anderson, D. R. (2004). Multimodel inference: understanding AIC and BIC in model selection. *Sociological Methods & Research*, *33*(2), 261–304.

College Board. (n.d.). *Accuplacer® sample questions for students*. Retrieved from https://accuplacer.collegeboard.org/students

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. T., & Anderson, J. R. (1990). The effect of feedback control on learning to program with the LISP Tutor. In M. Piattelli-Palmarini (Ed.), *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society* (pp. 796–803). Cambridge, MA: Lawrence Earlbaum Associates.

Corbett, A. T., & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, *4*(4), 253–278.

Falmagne, J.-C., Cosyn, E., Doignon, J.-P., & Thiéry, N. (2006). The assessment of knowledge, in theory and in practice. In *Formal concept analysis* (Vol. 3874, pp. 61–79). Dresden, Germany: Springer-Verlag, Berlin Heidelberg.

Falmagne, J.-C., & Doignon, J.-P. (2011). *Learning spaces: Interdisciplinary applied mathematics*. New York: Springer-Verlag.

Gobert, J. D., Sao Pedro, M., Raziuddin, J., & Baker, R. S. (2013). From log files to assessment metrics: Measuring students' science inquiry skills using educational data mining. *Journal of the Learning Sciences*, *22*(4), 521–563.

González-Brenes, J., Huang, Y., & Brusilovsky, P. (2014). General features in knowledge tracing to model multiple subskills, temporal item response theory, and expert knowledge. In J. Stamper, Z. Pardos, M, Mavrikis, & B. M. McLaren (Eds.), *The Proceedings of the 7th international conference on educational data mining* (pp. 84–91). London: International Educational Data Mining Society.

Griffiths, T. L., Chater, N., Kemp, C., Perfors, A., & Tenenbaum, J. B. (2010). Probabilistic models of cognition: Exploring representations and inductive biases. *Trends in Cognitive Sciences*, *14*(8), 357–364.

Heffernan, N. T., & Heffernan, C. L. (2014). The ASSISTments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, *24*(4), 470–497.

Kim, Y. J., Almond, R. G., & Shute, V. J. (2016). Applying evidence-centered design for the development of game-based assessments in physics playground. *International Journal of Testing*, *16*(2), 142–163.

Kinnebrew, J. S., Loretz, K. M., & Biswas, G. (2013). A contextualized, differential sequence mining method to derive students' learning behavior patterns. *JEDM - Journal of Educational Data Mining*, *5*(1), 190–219.

Koedinger, K. R., & Aleven, V. (2007). Exploring the assistance dilemma in experiments with cognitive tutors. *Educational Psychology Review*, *19*(3), 239–264.

Koedinger, K. R., Anderson, J., Hadley, W., & Mark, M. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education*, *8*, 30–43.

Koedinger, K. R., & MacLaren, B. A. (2002). *Developing a pedagogical domain theory of early algebra problem solving*, (Tech. Rep. No. CMU-HCII-02-100). Pittsburgh, PA: Carnegie Mellon University.

Koedinger, K. R., Pavlik, Jr., P. I., Stamper, J., Nixon, T., & Ritter, S. (2011). Avoiding problem selection thrashing with conjunctive knowledge tracing. In C. Conati, S. Ventura, M. Pechenizkiy, & T. Calders (Eds.), *Proceedings of the 4th International Conference on Educational Data Mining*. Eindhoven, Netherlands: International Educational Data Mining Society.

Kramer, A. (2010–2014). *Symja library – Java symbolic math system*. Available at: https://bitbucket.org/axelclk/symja_android_library/

LaMar, M. M. (2017). Markov decision process measurement model. *Psychometrika*, 1–22.

Lee, M. D. (2011). How cognitive modeling can benefit from hierarchical Bayesian models. *Journal of Mathematical Psychology*, *55*(1), 1–7.

Lindsey, R. V., Shroyer, J. D., Pashler, H., & Mozer, M. C. (2014). Improving students' long-term knowledge retention through personalized review. *Psychological Science*, *25*(3), 639–647.

Linn, M. C., Lee, H.-S., Tinker, R., Husic, F., & Chiu, J. L. (2006). Teaching and assessing knowledge integration in science. *Science*, *313*(5790), 1049–1050. https://doi.org/10.1126/science.1131408

MacLellan, C. J., Harpstead, E., Patel, R., & Koedinger, K. R. (2016). The apprentice learner architecture: Closing the loop between learning theory and educational data. In T. Barnes, M. Chi, & M, Feng (Eds.), *Proceedings of the 9th International Conference on Educational Data Mining* (pp. 151–158). Raleigh, NC: International Educational Data Mining Society.

Mattern, K. D., & Packman, S. (2000, December). *Predictive validity of accuplacer scores for course placement: A meta-analysis* (Tech. Rep.). College Board.

Matuk, C. F., Linn, M. C., & Eylon, B.-S. (2015). Technology to support teachers using evidence from student work to customize technology-enhanced inquiry units. *Instructional Science*, *43*(2), 229–257.

Nicaud, J., Chaachoua, H., & Bittar, M. (2006). Automatic calculation of students' conceptions in elementary algebra from Aplusix log files. In M. Ikeda, K. D. Ashley, & T. W. Chan (Eds.), *Intelligent tutoring systems. ITS 2006.*, Lecture Notes in Computer Science (Vol. *4053*, pp. 433–442). Berlin: Springer.

Pane, J. F., Griffin, B. A., McCaffrey, D. F., & Karam, R. (2014). Effectiveness of cognitive tutor algebra I at scale. *Educational Evaluation and Policy Analysis*, *36*(2), 127–144.

Payne, S., & Squibb, H. (1990). Algebra mal-rules and cognitive accounts of error. *Cognitive Science*, *14*(3), 445–481.

Rafferty, A. N., & Griffiths, T. L. (2015). Interpreting freeform equation solving. In C. Conati, N. Heffernan, A. Mitrovic, & M. F. Verdejo (Eds.), *Proceedings of the 17th International Conference on Artificial Intelligence in Education*. Lecture Notes in Artificial Intelligence (Vol. 9112, pp. 387–397). Geneva, Switzerland: Springer.

Rafferty, A. N., Jansen, R. A., & Griffiths, T. L. (2016). Using inverse planning for personalized feedback. In T. Barnes, M. Chi, & M. Feng (Eds.), *Proceedings of the 9th International Conference on Educational Data Mining* (pp. 472–477). Raleigh, NC: International Educational Data Mining Society.

Rafferty, A. N., LaMar, M. M., & Griffiths, T. L. (2015). Inferring learners' knowledge from their actions. *Cognitive Science*, *39*(3), 584–618.

Rafferty, A. N., Zaharia, M., & Griffiths, T. L. (2014). Optimally designing games for behavioural research. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *470*, 20130828.

Razzaq, L. M., Heffernan, N., Feng, M., & Pardos, Z. (2007). Developing fine-grained transfer models in the assistment system. *Journal of Technology, Instruction, Cognition, and Learning*, *5*(3), 289–304.

Ritter, S., Anderson, J. R., Koedinger, K. R., & Corbett, A. (2007). Cognitive Tutor: Applied research in mathematics education. *Psychonomic Bulletin & Review*, *14*(2), 249–255.

Schooler, L. J., & Anderson, J. R. (1990). The disruptive potential of immediate feedback. In M. Piattelli-Palmarini (Ed.), *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society* (pp. 702–708). Cambridge, MA: Lawrence Earlbaum Associates.

Shute, V. (2008). Focus on formative feedback. *Review of Educational Research*, *78*(1), 153–189.

Siegler, R. S., & Pyke, A. A. (2013). Developmental and individual differences in understanding of fractions. *Developmental Psychology*, *49*(10), 1994–2004.

Siegler, R. S., Thompson, C. A., & Schneider, M. (2011). An integrated theory of whole number and fractions development. *Cognitive Psychology*, *62*(4), 273–296.

Sleeman, D. (1984). An attempt to understand students' understanding of basic algebra. *Cognitive Science*, *8* (4), 387–412.

Sleeman, D., Kelly, A. E., Martinak, R., Ward, R. D., & Moore, J. L. (1989). Studies of diagnosis and remediation with high school algebra students. *Cognitive Science*, *13*(4), 551–568.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.

Tenenbaum, J. B., Kemp, C., Griffiths, T. L., & Goodman, N. D. (2011). How to grow a mind: Statistics, structure, and abstraction. *Science*, *331*(6022), 1279–1285.

Supporting Information

Additional supporting information may be found online in the Supporting Information section at the end of the article:

Appendix S1. Supplementary Information.

Figure S1. Performance and satisfaction with the website for participants in Experiment 1. (a) Time spent on website by condition. Error bars show one standard error. (b) Performance on website by condition. "Attempted" problems are those where the user entered at least one step; correct indicates the final answer was mathematically correct. (c) Website usability ratings.

Figure S2. Robustness to mastery cutoff of differential improvement based on feedback about mastered versus unmastered skill. (a) p-value for interaction between whether feedback was on a mastered skill and time of test for ANOVA predicting test score. The p-value is less than .05 for mastery cutoffs that are neither too high nor too low. (b) Effect size (partial η2) for the same interaction term.