International Journal of Artificial Intelligence in Education (2020) 30:616–636
https://doi.org/10.1007/s40593-020-00218-y

ARTICLE

Intelligent Feedback on Hypothesis Testing

Sietske Tacoma[1] · Bastiaan Heeren[1,2] · Johan Jeuring[1,2] · Paul Drijvers[1]

Accepted: 17 September 2020 / Published online: 9 October 2020

Abstract

Hypothesis testing involves a complex stepwise procedure that is challenging for many students in introductory university statistics courses. In this paper we assess how feedback from an Intelligent Tutoring System can address the logic of hypothesis testing and whether such feedback contributes to first-year social sciences students' proficiency in carrying out hypothesis tests. Feedback design combined elements of the model-tracing and constraint-based modeling paradigms, to address both the individual steps as well as the relations between steps. To evaluate the feedback, students in an experimental group ($N = 163$) received the designed intelligent feedback in six hypothesis-testing construction tasks, while students in a control group ($N = 151$) only received stepwise verification feedback in these tasks. Results showed that students receiving intelligent feedback spent more time on the tasks, solved more tasks and made fewer errors than students receiving only verification feedback. These positive results did not transfer to follow-up tasks, which might be a consequence of the isolated nature of these tasks. We conclude that the designed feedback may support students in learning to solve hypothesis-testing construction tasks independently and that it facilitates the creation of more hypothesis-testing construction tasks.

Keywords Feedback · Hypothesis testing · Intelligent tutoring systems · Statistics education

Introduction

Hypothesis testing is widely used in scientific research, and is therefore covered in most introductory statistics courses in higher education (Carver et al. 2016). This topic is challenging for many students, because it requires the ability to follow a complex line

✉ Sietske Tacoma
s.g.tacoma@uu.nl

1 Utrecht University, Utrecht, The Netherlands

2 Open University of the Netherlands, Heerlen, The Netherlands

of reasoning involving uncertainty (Falk and Greenbaum 1995; Garfield et al. 2008). Additionally, this line of reasoning involves several complex concepts, such as significance level, test value and p value (Castro Sotos et al. 2007). Students struggle to understand the role and interdependence of these concepts in the hypothesis testing procedure, or, in other words, the logic of hypothesis testing (Vallecillos 1999). Appropriate feedback could support students in comprehending this logic, by focusing the student's attention to currently relevant aspects and thus reducing cognitive load (Shute 2008). To address students' reasoning regarding the logic of hypothesis testing, feedback should address all aspects of a (partial) solution: not only the content of a current step, but also its relations to earlier steps.

Since groups in introductory statistics courses are often large, it is difficult for teachers to provide such sophisticated feedback to individual students. Computer-based learning environments could offer a solution: many are available for statistics education and the provision of feedback is seen as one of their most important potential benefits (Sosa et al. 2011). Examples of computer-based learning environments that provide feedback on hypothesis-testing tasks include ALEKS[1] and WISE[2] (Aberson et al. 2003). Many of these systems provide excellent explanations of the logic of hypothesis testing, often illustrated with interactive simulations. Yet, support for students in carrying out hypothesis tests by themselves tends to be limited. To enable provision of feedback, the separate steps of the hypothesis-testing procedure are often addressed in separate items, each with their own interaction components and feedback. While this approach offers students the opportunity to practice with the steps of the hypothesis-testing procedure, it provides little opportunity to reason about the logic of hypothesis testing and to reflect on the relations between the various steps.

A type of computer-based learning environment that does have the potential to provide feedback on the student's reasoning in hypothesis testing is the Intelligent Tutoring System (ITS). Many ITSs offer tasks in which students can construct multi-step solutions. Like human tutors, ITSs can provide feedback on the level of steps and about the relations between steps, as well as detailed diagnostics of student errors (Nwana 1990). Some ITSs have been found to be as effective as human tutors and, generally, ITSs that provide feedback on the level of steps have been found to be more effective than ITSs that provide feedback on the level of complete solutions (VanLehn 2011). However, ITSs are highly domain dependent and while ITSs have been designed for the domain of hypothesis testing (Kodaganallur et al. 2005), to our knowledge no critical evaluations of their effectiveness for learning have been reported up to date.

This paper, therefore, describes the conceptual design, implementation and evaluation of an ITS for hypothesis testing, in which students can construct hypothesis tests and receive feedback on their steps and reasoning. The contribution of this paper is twofold. First, to address the content of individual steps as well as the relations between steps, our ITS combines elements of two prevailing paradigms in ITS design: model tracing (Anderson et al. 1995) and constraint-based modeling (Mitrovic et al. 2007). Although our ITS is not the first to combine these two paradigms (e.g., Goguadze and Melis 2009; Roll et al. 2010), we contribute to the ongoing exploration of how these

[1] https://www.aleks.com

[2] wise.cgu.edu

two paradigms can strengthen each other in providing support to students who are constructing multistep solutions. The research question relating to this combination of paradigms is:

> RQ1: How can model tracing and constraint-based modeling be combined to generate feedback on students' reasoning in hypothesis-testing construction tasks?

Second, to stimulate students to set up hypothesis tests by themselves, the ITS we describe offers an innovative approach for providing hypothesis-testing tasks in computer-based learning environments. Instead of addressing separate steps in separate items, the student can freely select and order steps, while the ITS keeps an eye on the internal consistency of the student's solution so far. To evaluate the effects of the intelligent feedback within these innovative tasks, the following research question is addressed:

> RQ2: Does automated intelligent feedback addressing students' reasoning in hypothesis-testing construction tasks contribute to student proficiency in carrying out hypothesis tests?

To address these two research questions, this paper is organized as follows. In the next section, we provide a literature overview explaining the two paradigms for ITS design and showing how they have been combined before. Next, we discuss how the two paradigms could address typical fallacies in students' logical reasoning in hypothesis-testing construction tasks. After these two theoretical sections, we turn to the evaluation study, by describing the design and implementation of our ITS, the study design, results, and finally our conclusion and discussion regarding the research questions.

Related Work on Stepwise Feedback in ITS

Although ITSs vary considerably in design, they generally contain the following four components: an expert knowledge model, a student model, a tutoring model, and a user interface model (Nwana 1990; Woolf 2009). Of these four, the expert knowledge model is mainly responsible for diagnosing errors in student solutions and is, hence, highly domain-dependent. It contains information about domain knowledge required to solve tasks in the domain (Heeren and Jeuring 2014) and is therefore also regularly called the domain model or domain module (Woolf 2009). We briefly discuss the two paradigms for the design of domain modules that are combined in this study: model tracing and constraint-based modeling.

In the model-tracing approach, the ITS checks whether a student follows the rules of a model solution (Anderson et al. 1995). The domain module contains a set of expert rules, which an expert would apply to solve tasks in the domain. It may also contain buggy rules: incorrect rules reflecting incorrect domain knowledge. Finally, besides these static rules, it contains a reasoning engine in the form of a model tracer. This model tracer that can identify which expert and buggy rules a student has applied to arrive at a (partial) solution. A student's step is marked as an error if it either does not match any expert rule, or matches a buggy rule (Mitrovic et al. 2003). Furthermore, model-tracing domain modules can provide hints for appropriate next steps.

Constraint-based modeling concentrates on solutions, rather than on the solution process. The underlying idea is that incorrect knowledge emerges as inconsistencies in students' solutions (Mitrovic et al. 2007). Domain knowledge is represented as a set of constraints, consisting of a relevance condition and a satisfaction condition. Errors in student solutions emerge as violated constraints, that is, constraints for which the relevance condition is satisfied, but the satisfaction condition is not. If a student's solution does not violate any constraints, it is diagnosed as correct. If a student's solution violates multiple constraints at the same time, feedback messages for all violated constraints can be reported to the student, or a constraint prioritization can be used to decide which constraint should be handled first.

ITSs that support hypothesis testing have been designed based on either of these approaches (Kodaganallur et al. 2005). In their comparison, Kodaganallur and colleagues concluded that their model-tracing tutor could provide more targeted, high-quality remediation, but also that this had required greater development effort than building their constraint-based tutor. We do not believe that one or the other is a superior paradigm, but rather concur with Mitrovic et al. (2003) that both have their strengths and weaknesses. A strength of the constraint-based modeling approach that Mitrovic and colleagues mentioned is its flexibility to accommodate many different problem-solving strategies, while a strength of the model-tracing approach is its capability to provide targeted hints for specific strategies. Both strengths, however, are believed to be achievable in the other paradigm as well; they mainly seem to be *more straightforward* to achieve in the one paradigm than in the other.

Because both paradigms have their strengths, combining the paradigms could lead to useful tutoring and feedback. Two ITSs that combine both approaches are ActiveMath (Goguadze and Melis 2009) and the Invention Lab (Roll et al. 2010). In ActiveMath the correctness of students' answers is checked using constraints, while hints for next steps are generated using a model-tracing approach. In the Invention Lab, domain knowledge is modeled using constraints, while model tracing is used to provide feedback on the students' domain-independent inquiry strategies. Generally speaking, in both of these ITSs, constraints are used to check whether the domain-specific content of the solution is correct, while model-tracing elements deal with the sequencing of the steps in the solution process.

This sequencing of steps is an essential element of the hypothesis-testing procedure, and, as argued before, an element that students struggle with. Meanwhile, students also struggle with the many concepts that play a role in the hypothesis-testing procedure, and constraints may provide a more straightforward method to model the domain knowledge related to these concepts. From a theoretical perspective, therefore, combining both paradigms could result in useful feedback on students' logical reasoning in hypothesis tests. To further explore this potential, in the next section we discuss two examples of typical student errors in carrying out hypothesis tests, and the ways in which the two paradigms could diagnose such errors.

Stepwise Feedback on Hypothesis Testing

Feedback typically signals a gap between a student's current performance and desired performance, the feedback-standard gap (Kluger and DeNisi 1996). In the case of hypothesis testing, a feedback-standard gap can manifest itself in two ways:

- Missing information, such as a solution that contains a value for the test statistic, but no hypotheses to test;
- Inconsistent information, such as a right-sided rejection region for a left-sided test.

Model tracing and constraint-based modeling typically approach these gaps in different ways, which we illustrate with two examples.

The first example concerns a student who starts the solution process with calculating a value of the test statistic, without stating hypotheses. Although technically possible, from a pedagogical perspective this step is not desirable, because the meaning and interpretation of a value of the test statistic depend on the hypotheses that are tested. A constraint-based tutor, on the one hand, typically contains constraints that check for necessary elements in the solution (Mitrovic et al. 2003). For hypothesis testing, such a constraint could have relevance condition “the solution contains a value of the test statistic” and satisfaction condition “the solution contains hypotheses”. This example is one specific situation in which this constraint would be violated, but this constraint covers many more such situations. A feedback message corresponding to this constraint could encourage the student to first formulate hypotheses before proceeding with carrying out the test. A model-tracing tutor, on the other hand, would contain a rule for adding hypotheses as well as a rule for calculating the value of the test statistic. In this example, adding hypotheses would be an expected step, whereas calculating the value of the test statistic would not. Depending on the implementation, the student’s step of calculating the test statistic could be recognized as a detour from the expert strategy and this could be given as feedback to the student. Modeling exactly in which cases the step of adding the test statistic is an expected step and when it is a detour requires modeling many different solution states. Hence, providing explicit feedback about missing elements of a (partial) solution is possible in both paradigms, but may be more straightforward in the constraint-based paradigm.

The second example concerns inconsistent information in a solution. Suppose a student has almost finished a task: the hypotheses, critical value, rejection region and value of the test statistic comprise a logical line of reasoning. In the final step, however, the student draws an incorrect conclusion about the hypotheses. If the correct answer would be to reject the null hypothesis, then two conceptually different incorrect conclusions are possible: “Do not reject the null hypothesis” and “Accept the alternative hypothesis”. The first reflects an inconsistency between the previous steps and the final conclusion, while the second concerns a misunderstanding of the convention in hypothesis testing to draw conclusions about the null hypothesis and not about the alternative hypothesis. In a constraint-based tutor, these two pieces of domain knowledge could be captured in two constraints. The first would have relevance condition “the test statistic lies inside the rejection region and a conclusion is drawn” and as satisfaction condition “the conclusion is to reject the null hypothesis”. This constraint is violated by both errors described above. The second constraint, addressing the convention, would have as relevance condition “a conclusion is drawn” and as satisfaction condition “the conclusion concerns the null hypothesis” and is only violated by the second incorrect answer. When constraints are defined this way, the prioritization of constraints is important to distinguish between such errors. Alternatively, constraints could be defined at a more specific level, for example by adding “the conclusion concerns the null hypothesis” to the relevance condition of the first constraint. The model-tracing approach for this situation may be slightly more straightforward: a model-tracing tutor can contain buggy rules for each of the two error types and provide appropriate feedback for each one of them (Mitrovic et al. 2003).

To summarize, both the constraint-based and the model-tracing paradigm have their merits for addressing the logic of hypothesis testing. A final typical feature of model-tracing tutors is that they can express hints in terms of what a student needs for a logical next step in the current line of reasoning (Goguadze and Melis 2009). Advice from constraint-based tutors tends to focus more on desired features of the final solution than on the order in which these features are added to the solution (Mitrovic et al. 2003). Together, these two aspects can help students gain insight into the steps that are essential for hypothesis testing and the order in which they are typically carried out. From a pedagogical perspective, therefore, combining both paradigms into a single ITS for hypothesis testing seems promising. In the following sections we turn to a design study evaluating this combination, implemented in six hypothesis-testing construction tasks, in practice.

Methods

The study was designed as a randomized controlled experiment that was embedded in a compulsory course on Methods and Statistics for first-year psychology students at a Dutch university. In five weeks of this ten-week course students received online homework sets containing 7 to 13 tasks, which were designed in the Digital Mathematics Environment (Drijvers et al. 2013). The Digital Mathematics Environment supports various interaction types, such as formula input and multiple-choice items. It was turned into an ITS by adding a domain module, which enabled providing intelligent feedback on hypothesis-testing construction tasks.

Design of Hypothesis-Testing Construction Tasks

The third, fourth and fifth homework set concerned hypothesis testing. Each of these homework sets contained two tasks specifically aimed at developing the students' proficiency in carrying out hypothesis tests, by asking the students to select steps from a drop-down menu and to complete these steps. An example is shown in Fig. 1: after selecting a step from the drop-down menu called "Action", it appears as next step in the step construction area. Next, the student can complete the step by filling in the answer boxes and use the check button to check the procedure so far. Answer boxes are either number input boxes, in which students can fill in any number, or drop-down menus offering students two to six answer alternatives for (part of) the current step. After finishing the hypothesis-testing procedure, the student should state the overall conclusion in the final conclusion area below the drop-down menu with steps.[3]

Two versions of the homework sets were designed: an experimental version in which intelligent feedback on the steps in the hypothesis-testing procedure was provided by the ITS, and a control version that only provided verification feedback on the individual answer boxes in the steps. Hence, in the experimental condition students

[3] A more elaborate example can be found at https://www.youtube.com/watch?v=toXFJhFJI5w

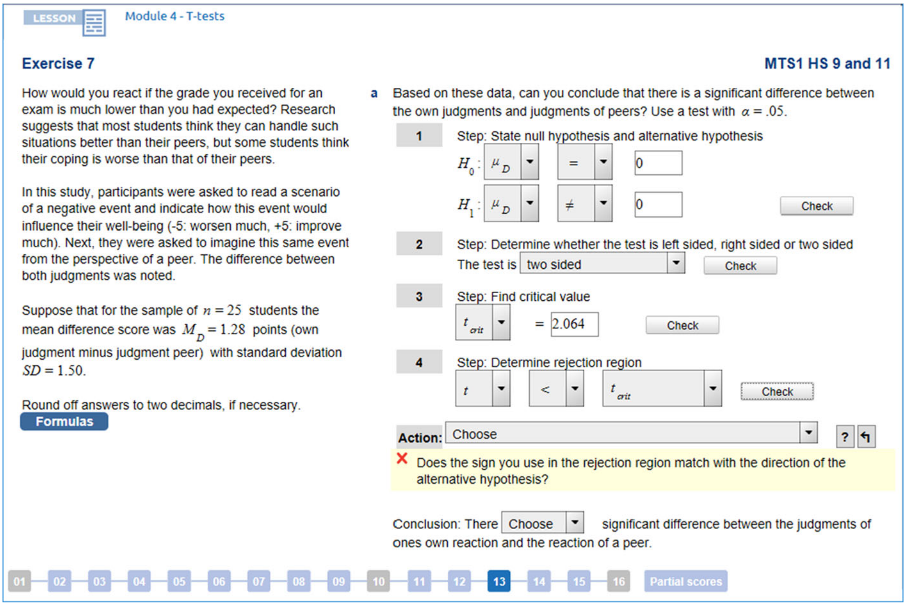

Fig. 1 Hypothesis-testing construction task in the Digital Mathematics Environment (translated)

received elaborate feedback on fallacies in the logic of their hypothesis tests, while in the control condition students only received feedback on the correctness of their current step, irrespective of previous steps. Figure 2 shows an enlarged version of the feedback in the experimental condition that is shown in Fig. 1. Figure 3 shows the feedback for the same partial solution in the control condition. This example illustrates how the ITS feedback addresses the student's error in relation with the statistical concepts involved, while in the control condition the error is only flagged, without further elaboration. A second difference between the two versions is the availability of a hint button in the experimental version (the button with the question mark in Figs. 1 and 2). Apart from the six hypothesis-testing construction tasks, all tasks in the two versions were equal.

Design of the Domain Module

The technical design of the domain module is based on the Ideas framework (Heeren and Jeuring 2014), which uses a model-tracing approach to calculate feedback and hints. For this study, this framework was expanded to also support constraints. The final domain module contains 36 expert rules, 16 buggy rules, and 49 constraints. Table 1 presents examples of an expert rule, a buggy rule, and a constraint, each with their corresponding feedback messages.

The design of expert rules, buggy rules and constraints was informed by discussions with four teachers of introductory university statistics courses about the logic of hypothesis testing and common errors by students. Furthermore, textbooks were consulted. Based on this input, we decided to support two methods for logical reasoning in carrying out a hypothesis test: the conclusion about the hypotheses can be drawn based on comparison of the test statistic with a critical value, or based on comparison of a p value with a significance level. In each method, a complete solution should include four essential steps: (1) state hypotheses,

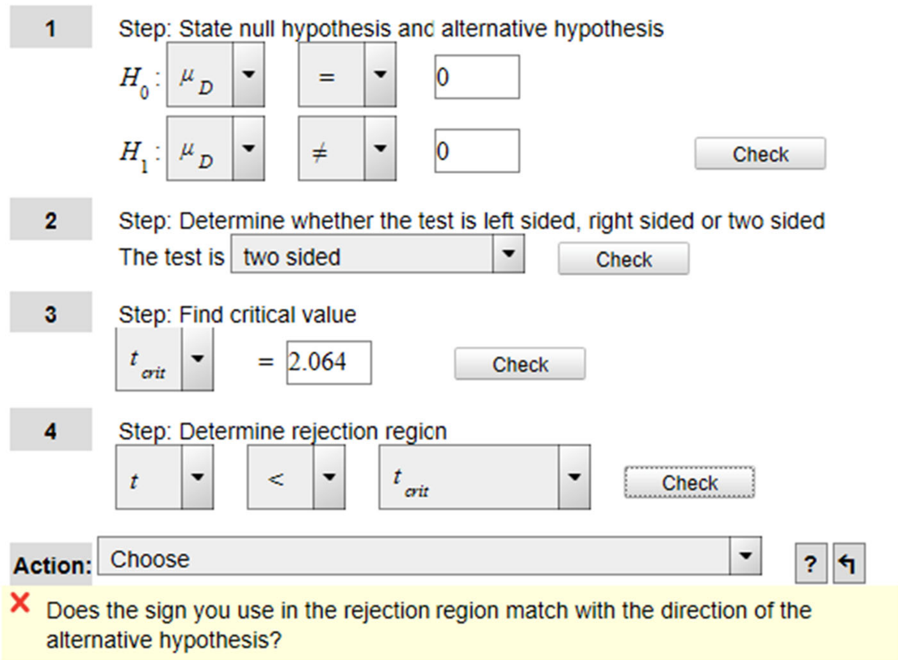

Fig. 2 Example of feedback in the experimental condition

(2) calculate a test statistic, (3) either find a critical value or find a p value, and (4) draw a conclusion about the hypotheses. Although crucial for the logic of hypothesis testing, stating a significance level and selecting an appropriate statistical test were not regarded as essential

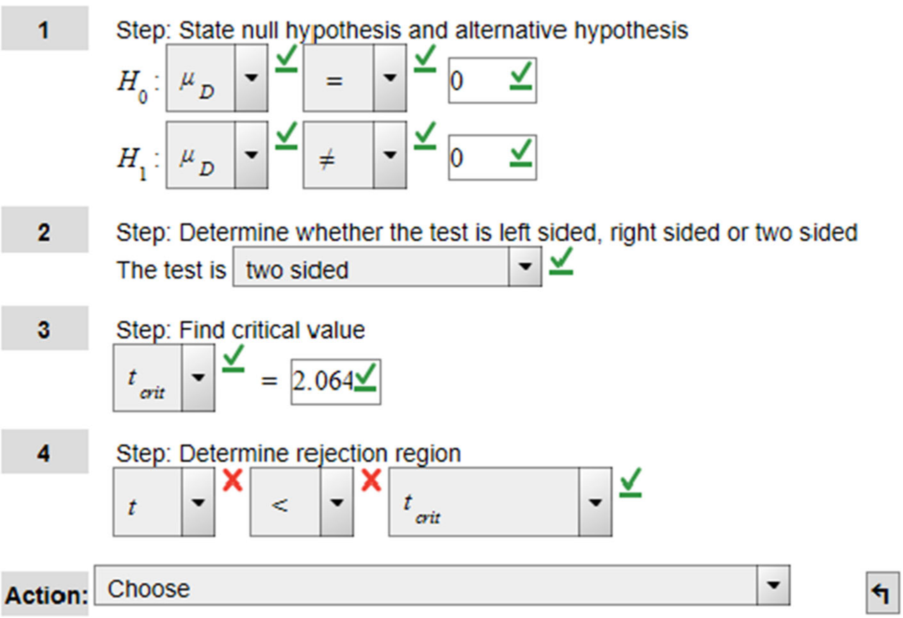

Fig. 3 Example of feedback in the control condition, for the same partial solution as in Fig. 2

Table 1 Examples of an expert rule, a buggy rule and a constraint in the designed domain module

Component	Example	Feedback message
Expert rule	**If** the alternative hypothesis contains a ≠−sign **Then** add a two-sided rejection region	This is the correct rejection region for this test.
Buggy rule	**If** objects in samples can be paired **Then** add a t-test for independent groups as test type	A t-test for independent groups is appropriate if the objects in the two samples cannot be paired.
Constraint	**Relevance condition**: The solution contains a rejection region **Satisfaction condition**: The solution contains an alternative hypothesis	To which hypotheses does this rejection region correspond? First state hypotheses.

steps, because they were specified in all task descriptions. Besides these essential steps, students could include several other steps, such as a summary of sample statistics and a specification of whether the test was left-sided, right-sided or two-sided.

Since the domain module required these essential steps, assessment criteria for correct solutions were stricter in the experimental condition than in the control condition. Where in the experimental condition correct solutions needed to include four essential steps, in the control condition students only needed to include a correct conclusion about the null hypothesis for a solution to be correct. Besides rules, buggy rules and constraints, the domain module also contains a reasoning engine that reasons with these rules and constraints. This engine contains two components that deal with prioritizing rules and constraints in cases where more than one is applicable: a rule ordering and a constraints prioritization.[4] Furthermore, it contains knowledge of how rules and constraints can be combined to diagnose students' solutions. Figure 4 illustrates the reasoning engine's checking procedure, which results in a diagnosis about a student's current solution. This checking procedure is carried out each time a student adds a step – such as defining an alternative hypothesis or calculating the value of a test statistic – in a hypothesis-testing construction task.

First, all constraints are checked. The constraints are assumed to be complete, which means that together they separate consistent from inconsistent (partial) solutions: a partial solution is consistent if and only if it does not violate any constraint. If a solution violates one or more constraints, the reasoning engine determines whether a buggy rule was applied. Through this structure of buggy rules following constraints, buggy rules could be used to zoom in on the student's error. Consider, for example, a student who selected a t-test for independent groups where a t-test for dependent groups would be appropriate. A constraint concerning the test type is violated, but (in our implementation) the same constraint would have been violated when the student had selected a z-test or t-test for one group. In this situation, we chose to use buggy rules to distinguish between these conceptually different errors and provide error specific feedback messages like the example in Table 1.[5] For partial solutions in which no buggy rule was

[4] The most recent version of the domain module software is available at http://hackage.haskell.org/package/ideas-statistics

[5] Another way to resolve this issue is to define separate, more specific, constraints for the various situations

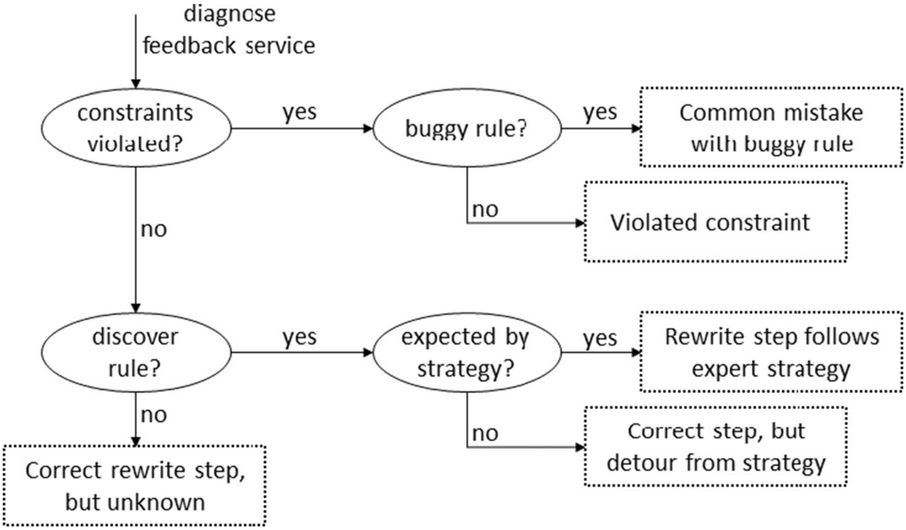

Fig. 4 The reasoning engine's diagnose feedback service

applied, a general message for the violated constraint is reported. For example, a partial solution that contains a rejection region but no alternative hypothesis violates the constraint presented in Table 1. The feedback message for this constraint addresses the role of the hypotheses in the hypothesis-testing procedure, thus drawing attention to the logic of hypothesis testing. When more than one constraint is violated, the constraints prioritization determines for which constraint a feedback message is displayed.

If no constraints are violated, there is no need to check the buggy rules, because of the completeness of the constraints: if a buggy rule was applied, then at least one constraint would have been violated as well. Therefore, the reasoning engine only needs to attempt to discover which rule the student has applied to arrive at the current partial solution. If no rule is identified, the step taken by the student is marked as a correct but unknown. This is an advantage of the constraints structure: students can add multiple steps at once and, as long as no constraints are violated, this is regarded a correct solution path. In a tutor based solely on model tracing, to allow adding multiple steps at once all possible combinations of steps would have to be checked, which requires considerable computing power when many steps are applicable at once. If the reasoning engine does identify a rule that the student has applied, it checks whether this is an expected rule in the expert strategy, so that a detour from this strategy can be signaled. In the implementation in this study, though, no distinction was made between rules following the strategy and not following the strategy. In both cases, a feedback message for the identified rule is displayed, such as the example in Table 1. Besides checking partial solutions, the reasoning engine can also provide hints on next steps to take, by identifying a rule that would be appropriate to apply for the current partial solution. This feature, enabled by the model-tracing basis of the domain module, could also be used to generate a worked-out solution. In this study, though, the possibility of worked-out solutions was not exploited.

To identify and resolve technical flaws and unclarities in the design of the tasks and the domain module, a first version was piloted with five students. After the pilot, several improvements were made to feedback formulation and prioritization of rules and constraints.

Participants

Participants in this study, the first-year psychology students enrolled in the Methods and Statistics course, were divided randomly into an experimental and a control group. From the 310 students in the experimental group 226 students worked on the hypothesis-testing construction tasks, of which 163 gave consent for the use of their work in this study. From the 309 students in the control group 216 students worked on the tasks, of which 151 gave consent. The participants were between 17 and 31 years old and student age did not differ significantly between groups ($M = 19.4$ years, $SD = 1.6$ years in the experimental group, $M = 19.2$ years, $SD = 1.9$ years in the control group, $t(312) = 1.22$, $p = .222$). The majority of students (77%) was female and this percentage did also not differ significantly between groups (75% in the experimental group, 78% in the control group, $X^2(1, N = 314) = 0.18$, $p = .668$). The students' statistical ability, as measured by an intermediate statistics exam that was administered before the third week of the course, also did not differ significantly between groups ($M = 4.73$ points (out of 7), $SD = 1.28$ in the experimental group, $M = 4.61$ points, $SD = 1.24$ in the control group, $t(300) = 0.77$, $p = .439$). To reduce research participation effects, i.e. students possibly behaving differently because they were part of an experiment (McCambridge et al. 2014), the students, both in the experimental and the control group, were not given all information: they were told that they were part of an experiment and asked for their consent, but they were not told about the different conditions and which condition they were assigned to.

Data Collection and Analysis

Data for this study consisted of logs of the students' actions on the online homework sets. These logs included all attempts students made to construct correct answers to the tasks, and all feedback and hint requests. After exporting the logs from the Digital Mathematics Environment, logs from students who did not give consent were deleted and all other logs were anonymized.

Data analysis focused on three aspects of the students' work:

A1. The amount of work done by students in the ITS feedback condition and the control condition, and the amount of feedback they received on the six hypothesis-testing construction tasks;

A2. Performance on the six hypothesis-testing construction tasks, as measured by (1) number of tasks attempted, (2) number of tasks solved, and (3) number of errors concerning the logic of hypothesis testing;

A3. Performance on follow-up tasks about hypothesis testing without intelligent feedback.

The first aspect, A1, was deemed relevant, because students can only learn from feedback if they indeed receive it. And to receive feedback, students need to work on the tasks. The time students worked on the tasks and mean number of steps students selected were compared between groups. Since samples were large (more than 100 students in each group), independent samples *t*-tests were used for all comparisons between groups (Field 2009). Welch two sample *t*-tests were used when variances were not equal in both groups, as tested by Levene's test. Furthermore, for students in the experimental group the number of feedback messages received and hints requested were calculated per task.

Regarding A2, three measures were used to assess student performance on the six tasks: (1) number of tasks in which students attempted to construct steps, (2) number of tasks that students solved completely, and (3) number of errors students made concerning the logic of hypothesis testing. The first measure (A2, measure 1) was regarded as indicator of feedback effectiveness, since the ITS feedback was designed to support students in the step construction process. Students who did not attempt to construct steps in later tasks apparently did not perceive the feedback on steps in earlier tasks as helpful (Narciss et al. 2014). While the more elaborate feedback by the ITS was expected to encourage students to attempt constructing steps, at the same time it required students to include steps in a correct order, which could lead to frustration and giving up on tasks.

Since the feedback was intended to contribute to the students' ability to solve the tasks, the number of solved tasks (A2, measure 2) is also an indicator of feedback effectiveness (Narciss et al. 2014). Students' solutions in the control group were assessed twice: according to their own group's criterion of stating a correct conclusion about the null hypothesis and according to the experimental group's criterion of including all four essential steps. Due to the intelligent feedback, students in the experimental group were expected to solve more tasks than students in the control group. Due to the difference in assessment criteria, however, students in the control group could be expected to solve more tasks under their own assessment criteria than students in the experimental group. The comparison between groups with a *t*-test was complemented with a logistic multilevel regression model (Hox et al. 2018) to assess the progression of the difference between groups over time. A multilevel regression analysis was deemed most appropriate given the structure of the data: for each student there were up to six observations of solved or non-solved tasks. The lowest level of the multilevel model was, therefore, the task level and the highest level was the student level. The regression model was built in the software program HLM using full maximum likelihood estimation, as described in Hox et al. (ibid.).

The final measure of student performance on the six tasks was the number of errors that students made in the logical reasoning of their hypothesis tests (A2, measure 3). The ITS was especially designed to provide students with feedback about the logic of hypothesis testing, that is, the order of and relations between steps. The number of errors concerning this logic was expected to decrease over time in both groups, but more strongly in the experimental than in the control group. To assess the evolution of the difference between groups over time, we employed a *t*-test and a multilevel regression model (Hox et al. 2018).

Concerning A3, we notice that promising effects of feedback on student performance on the tasks for which feedback is provided do not automatically guarantee transfer to new tasks (Shute 2008). We therefore also assessed student performance on follow-up tasks about hypothesis testing, in which no intelligent feedback was provided. The online homework sets contained 31 follow-up sub-tasks on hypothesis testing. For all students who received feedback on constructed steps at least once the ratio between the number of these 31 sub-tasks that they answered correctly on their first attempt and the number of sub-tasks they attempted was calculated and these ratios were compared between groups.

Results

Results on A1: Summary of Steps Done and Feedback Received

Table 2 summarizes the average number of steps that students in both groups made and the number of feedback messages and hints students in the experimental group received. Students in the experimental group made slightly but significantly more steps ($M = 8.0$, $SD = 5.4$) than students in the control group ($M = 6.7$, $SD = 3.9$, $t(293.7) = 2.41$, $p = .016$, Cohen's $d = 0.27$). This is also reflected in the total time students worked on the six hypothesis-testing construction tasks: in the experimental group this was 41 min ($SD = 27$ min) and in the control group it was 32 min ($SD = 19$ min), a significant difference ($t(291.8) = 3.41$, $p < .001$, Cohen's $d = 0.38$). In both groups the number of steps decreased over tasks. It should be noted that in the final two tasks the test statistic was given, so fewer steps were needed for a complete solution than in earlier tasks. Finally, the number of feedback messages per student in the experimental group is quite high, especially in the first two tasks, implying that students received feedback on a regular basis. Students also regularly made use of the hints, with an average of two hint requests per student per task.

Results on A2: Performance on Six Hypothesis-Testing Construction Tasks

The average number of tasks students worked on, i.e. tasks in which they filled in the final answer box, and the average number of tasks in which students tried to construct

Table 2 Steps in both groups and feedback messages and hints in experimental group

	Experimental group				Control group	
Task	N	Steps per student (SD)	Feedback messages per student (SD)	Hints per student (SD)	N	Steps per student (SD)
3.4	154	14.0 (10.3)	23.2 (21.2)	3.7 (7.5)	143	11.3 (7.9)
3.6	111	11.2 (6.7)	22.3 (23.3)	2.4 (4.6)	105	9.1 (6.9)
4.7	134	6.8 (6.7)	11.4 (13.4)	1.3 (4.2)	130	6.5 (5.8)
4.8	118	7.1 (6.2)	16.2 (23.1)	2.5 (5.0)	115	6.1 (5.5)
5.3	134	4.9 (5.6)	7.7 (14.2)	1.5 (4.0)	127	4.1 (5.0)
5.6	127	3.9 (4.8)	5.6 (7.7)	1.4 (3.7)	123	3.3 (4.0)
All	163	8.0 (5.4)	14.1 (12.7)	2.0 (3.5)	151	6.7 (3.9)

steps (A2, measure 1) are summarized in Table 3. In both groups, students attempted to construct steps using the drop-down menu for almost 80% of the tasks they worked on. For the other 20% of the tasks, students may have used other means than the stepwise construction area to solve the task, or may have collaborated with a peer. The numbers of tasks students worked on and attempted step construction for did not differ significantly between groups.

In Table 3, the third and fourth lines summarize the average number of tasks that students solved completely (A2, measure 2). Students succeeded in solving the task in approximately half of the cases in which they attempted to construct steps. Over all six tasks, students in the control group solved slightly more tasks than students in the experimental group. This could be a consequence of the stricter assessment criterion for complete solutions in the experimental group, which required students to include all essential steps in their solution. When assessed following this stricter criterion, the number of complete solutions in the control group dropped to an average of 1.4 per student. Over all six tasks together, these differences between groups were not significant, as the results in Table 3 show. Given that students started off with the same prior knowledge, however, differences between groups were expected to emerge over time. A logistic multilevel regression model was created to take this effect of time into account. The model is summarized in Table 4.

The baseline model in Table 4 only included task number as predictor for solving the task. It reveals that the probability of solving a task decreased with task number, meaning that, generally, for higher task numbers the proportion of students who solved the task decreased. Including ITS feedback availability (M2) did not significantly improve the model: the deviance change was 3.70, which, with one degree of freedom for one extra estimated parameter, results in a p value of .054. This aligns with our previous finding that over all tasks together ITS feedback availability did not make a difference for the number of tasks students solved. The explanatory power of M2 was slightly higher than that of M1, though. Especially, while M1 only predicted 51% of the solved tasks correctly, M2 predicted 60% correctly. The addition of an interaction effect between task number and ITS feedback availability (M3) improved the model further: the deviance change was 13.92, which, with one degree of freedom for one extra estimated parameter, results in $p < .001$, hence a significant improvement to the model. The regression equation for this final model is:

Table 3 Student results on the six hypothesis-testing construction tasks

	Experimental group ($N = 163$)	Control group ($N = 151$)	t (df = 312)	p
Tasks worked on	4.8 (1.5)	4.9 (1.5)	0.86	.391
Tasks tried constructing steps	3.8 (1.7)	3.9 (1.6)	0.62	.537
Tasks with complete solution	1.7 (1.8)	2.0 (1.7)	1.33	.184
Tasks with correct essential steps	1.7 (1.8)	1.4 (1.6)	−1.59	.113

Table 4 Logistic multilevel regression model predicting the probability of solving a task from task number, ITS feedback availability and their interaction

	M1: Baseline	M2: + condition	M3: + interaction condition/task
Predictor coefficients			
Intercept	0.23	0.46*	0.79***
Task number	−0.24***	−0.24***	−0.41***
ITS feedback		−0.43	−1.07***
ITS feedback × Task number			0.32***
Model fit			
Deviance	3762.80	3759.10	3745.18
Estimated parameters	3	4	5
Deviance change		3.70	13.92***
Explanatory power			
Proportion solved tasks predicted correctly	.51	.60	.60
φ correlation coefficient	.16	.17	.17

*$p < .05$; **$p < .01$; ***$p < .001$

$$logit\left(p_{ij}\right) = 0.79–0.41 \cdot (i–1)–1.07 \cdot ITS\ \ feedback_j + 0.32 \cdot (i–1) \cdot ITS\,feedback_j + u_{0j},$$

with:

p_{ij}	the estimated probability that student j solved task i correctly
$ITS\,feedback_j$	equal to 0 (control group) and 1 (experimental group),
i	representing the task number (between 1 and 6), and
u_{0j}	a residual variance term for student j.

As in the baseline model M1, the negative regression coefficient for task number in the final model indicates that the probability of solving tasks decreased for later tasks. Filling in $i = 1$ and taking the inverse logit shows that the estimated probability of solving the first task was on average $\text{logit}^{-1}(0.79) = 0.69$ in the control group and $\text{logit}^{-1}(0.79–1.07) = 0.43$ in the experimental group, showing that initially students in the experimental group had more difficulty solving the tasks than students in the control group. This could be a consequence of the stricter assessment criteria in the experimental group, which students needed to get used to. Finally, the coefficient for the interaction term between ITS feedback availability and task number is positive. Hence, while for students in the control group the logit decreased by 0.41 per task, for students in the experimental group it only decreased by 0.41–0.32 = 0.09 per task. This suggests that the ITS feedback more effectively supported students in persevering to solve tasks than the control feedback, even though the assessment criteria for their solutions were stricter. This is also reflected in Fig. 5 (left), which displays the percentage of students who found complete solutions to each task, as percentage of students who attempted constructing steps for each task. For the first three tasks the percentage was smaller for students in the experimental group than for students in the

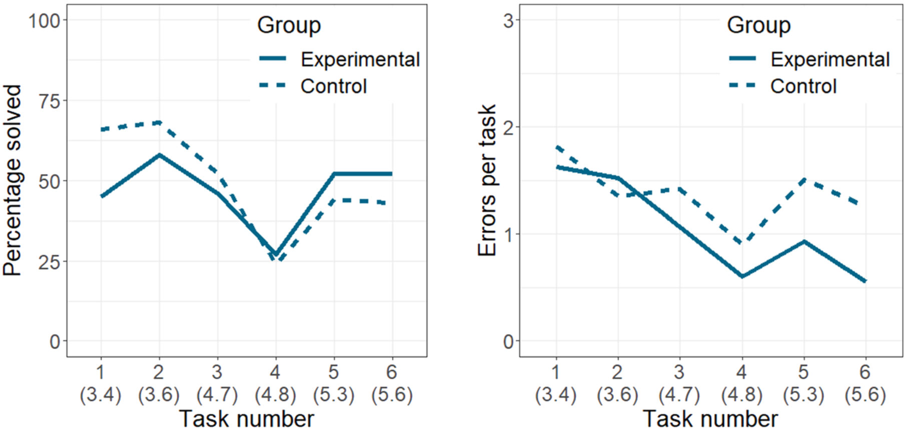

Fig. 5 Percentage of students who correctly solved tasks according to group's assessment criteria (left) and mean number of errors concerning the logic of hypothesis testing (right)

control group, but for the latter three tasks this was reversed. Hence, over time, students in the experimental group seemed to become relatively more proficient in solving hypothesis-testing construction tasks than students in the control group.

The final measure of student performance on the six tasks was the number of errors that students made in the logical reasoning of their hypothesis tests (A2, measure 3). The ITS could diagnose 15 different errors concerning hypothesis-testing logic, such as a missing alternative hypothesis. On average, students in the experimental group made 1.12 ($SD = 0.79$) different errors per solution, while students in the control group made 1.42 ($SD = 0.86$) different errors, which was significantly more ($t(312) = 3.22$, $p = .001$, Cohen's $d = 0.36$). The graph in Fig. 5 (right) displays the mean number of errors by students in both groups for each task. It shows that in both groups the number of errors decreased over tasks, but this trend was stronger in the experimental group. Fitting a multilevel regression model confirmed this impression. The resulting model is summarized in Table 5.

The baseline model (M1) included a linear and quadratic term for task number as predictors and showed that, generally, the number of errors decreased over time. The significance of the quadratic term suggests that the number of errors decreased quickly for the first tasks and more slowly for later tasks. In M2, ITS feedback availability was added to the baseline model, which resulted in a significantly better model fit ($p < .001$). The coefficient for ITS feedback availability was negative and significantly different from 0, confirming that the number of errors concerning hypothesis-testing logic was lower in the experimental group than in the control group. The variance at the student level decreased by 0.019, or 8.0% of the initial variance of 0.238. Hence, experimental condition explained 8% of the variance in number of errors per student. Adding the interaction effect between task number and ITS feedback availability (M3) again yielded a significantly better model fit ($p < .001$). In this model, the effect of ITS feedback availability itself became non-significant. This implies that for the first task, ITS feedback availability did not have a significant effect on the number of errors students made. Meanwhile, the significant interaction effect between ITS feedback

Table 5 Multilevel regression model predicting number of errors concerning hypothesis-testing logic from task number and task number squared, ITS feedback availability and interaction between task number and ITS feedback availability

	M1: Baseline	M2: + condition	M3: + interaction condition/task	M4: - condition
Fixed part				
Intercept	2.18***	2.33***	2.12***	2.17***
Task	−0.49***	−0.49***	−0.42***	−0.43***
Task quadratic	0.04***	0.05***	0.05***	0.05***
ITS feedback		−0.29***	−0.10	
ITS feedback × task			−0.13***	−0.11***
Random part				
σ_e^2	1.208	1.207	1.189	1.190
σ_{u0}^2	0.238	0.219	0.227	0.227
Model fit				
Deviance	3827.82	3816.12	3804.75	3805.22
Estimated parameters	4	5	6	5
Deviance change		11.70***	11.37***	−0.47

*$p < .05$; **$p < .01$; ***$p < .001$

availability and time implies that, over time, students in the experimental group made significantly fewer errors concerning the logic of their hypothesis tests than students in the control group. Removing the non-significant predictor ITS feedback availability (M4) yielded an equally good model – the deviance change is very small and not significant ($p = .493$) – with fewer estimated parameters. Comparing this model to the baseline model shows that the interaction between ITS feedback availability and task number explained 1.5% of the variance at task level and 4.6% of the variance at student level. In other words, the ITS feedback resulted in a slightly stronger decrease in number of errors for students in the experimental group than for students in the control group.

Results on A3: Transfer of Feedback Effects to Follow-up Tasks

Students in the experimental group ($N = 158$) and the control group ($N = 147$) performed similarly on the selection of follow-up hypothesis-testing tasks: the mean ratio of correct answers was 0.72 ($SD = 0.07$) in the experimental group and 0.71 ($SD = 0.08$) in the control group. The time that students worked on these tasks was also very similar in the experimental and control group: 49 min ($SD = 17$ min) for both groups. Hence, the effects of the ITS feedback did not transfer to the follow-up tasks that students were offered in the course. For comparison, though, we note that the mean ratio of immediately correct answers over all tasks that were identical in both groups (i.e. all tasks except for the six tasks concerning stepwise hypothesis testing) was 0.67 ($SD = 0.05$). Hence, compared to other tasks the students in both groups performed relatively well on the follow-up tasks on hypothesis testing.

Conclusion and Discussion

In this paper we have discussed the design, implementation and evaluation of an ITS that provides feedback on the logic of hypothesis testing, guided by two research questions. The first research question focused on how two design paradigms, model tracing and constraint-based modeling, could be combined to generate useful feedback regarding the logic of hypothesis testing. The second research questions concerned the effects of this feedback in hypothesis-testing construction tasks on the students' proficiency in carrying out hypothesis tests. After the ITS had been designed, students in an experimental and a control group worked on six hypothesis-testing construction tasks, in which they received a substantial amount of feedback and hints.

Regarding the first research question, we found constraints to be very useful for identifying missing elements and inconsistencies in students' solutions. This allowed the ITS feedback to address fallacies in the logic of the students' hypothesis tests. In our implementation, it also gave students the freedom to choose whether they wanted to check their solution after each single step, or after making several steps. Simultaneously, model-tracing elements allowed easily addressing specific common errors, for example related to the choice of the hypothesis test type. Model tracing was also helpful for providing hints on appropriate next steps in the student's line of reasoning. Combined, the paradigms allowed for diagnosing errors regarding most aspects of the logic of hypothesis testing that teachers and researchers had identified as challenging for students. Our division of labor between constraint-based and model-tracing elements resembles the division of labor in ActiveMath (Goguadze and Melis 2009). Where in ActiveMath, as well as in the Invention Lab (Roll et al. 2010) the two paradigms were used to reason about separate aspects of the domain, our design shows that they can also be integrated to cooperatively reason about the domain as a whole.

Concerning the second research question, we found that the ITS feedback did not seem to influence the number of tasks students attempted to construct steps in, but did seem to affect their success in solving tasks. Relatively fewer students in the experimental than in the control group solved the first three hypothesis-testing construction tasks, while for the later three tasks students in the experimental group persevered and succeeded better in solving the tasks. This suggests that after a period of familiarization with the ITS feedback students started to benefit from it. Furthermore, the number of errors students made in the logical reasoning of the hypothesis-testing procedure decreased significantly more over time for students receiving ITS feedback than for students receiving verification feedback only. Hence, the ITS feedback seemed to effectively support students in resolving their misunderstandings and, in this way, to contribute to student proficiency in carrying out hypothesis tests. Despite these promising results, no differences between groups were found in performance on follow-up tasks, which implies that there was no automatic transfer from the positive ITS feedback effects.

Although such a lack of transfer is often found (Shute 2008), in the case of this study it could be due to the design of the follow-up tasks. This was a limitation of the study: contrary to the six hypothesis-testing construction tasks, none of the follow-up tasks specifically addressed the logical reasoning in the hypothesis-testing procedure. Instead, the steps of the hypothesis-testing procedure were already given and students were only asked to fill in contents of individual steps. From a research perspective, availability of tasks addressing the logical reasoning could have provided more insight into transfer of the positive ITS feedback effects to other tasks. From an educational perspective, availability of such tasks would have been valuable as well, to avoid that students rely too much on the ITS feedback (Shute 2008).

A second limitation of this study was that, in spite of serious testing and pilots, in this first large-scale implementation of the ITS inevitably some unclarities and technical flaws became apparent. A small number of feedback messages provided incorrect or unsuitable information about current errors, and hints could only suggest a next step to take, regardless of whether the student's current partial solution was correct. For incorrect solutions, a hint containing guidance on how to resolve the current error would have been more appropriate. Nonetheless, the large collection of student data did provide a strong basis to inform improvements to the ITS' domain module, and especially for designing a hint structure that suits the combination of the model-tracing and constraint-based modeling approach. Furthermore, even though sometimes encountering confusing feedback messages and hints, students in general kept attempting to construct steps and, as the results above show, did still benefit from the feedback.

In spite of these limitations, combining the model-tracing and constraint-based paradigm to provide feedback on hypothesis-testing construction tasks seems to have resulted in a useful ITS for hypothesis testing. The ITS has not only supported students in solving more of the later tasks and making fewer errors in these tasks, but also to work significantly longer on the tasks and make significantly more steps. As Narciss et al. (2014) argue, doing more work may result in more opportunities to practice, meaning that the ITS feedback may stimulate students to engage more deeply with the concepts and logical reasoning involved in hypothesis testing. Finally, the finding that students in the experimental group made fewer errors in later tasks than students in the control group indicates that students became less and less dependent on the feedback for solving the tasks. This effect is in line with earlier findings for ITS feedback effectiveness (Steenbergen-Hu and Cooper 2014; Van der Kleij et al. 2015) and the effect size found in this study, Cohen's $d = 0.36$, is similar to those reported in Steenbergen-Hu and Cooper's review on the effectiveness of ITS feedback in higher education (Steenbergen-Hu and Cooper 2014).

Overall, this study suggests that combining the model-tracing and constraint-based modeling paradigms in an ITS for hypothesis testing is not only promising in theory, but also in educational practice. An additional aspect of this approach that is worth mentioning is that, albeit after a considerable initial design effort, it allows for easy adjustment of tasks to create new tasks. Once a start situation for a task is given, the ITS's model-tracing components can generate the solution and all steps towards the solution. This means that, contrary to the design in the control condition, the designer does not need to provide answers for all intermediate steps. Hence, even if the results do not transfer to follow-up tasks, with the ITS feedback available less design effort is needed for similar learning results. This invites the design of more tasks, offering students who need it more practice. In future designs, the ITS's potential for generating worked-out solutions, as well as the possibility to distinguish between expected steps and steps that deviate from the expected strategy, could be exploited further. Finally, a challenging aspect of hypothesis testing that is not yet addressed by the ITS feedback in this study is the role of uncertainty in the interpretation of the results from hypothesis tests (Falk and Greenbaum 1995). Future research could focus on broadening the scope of the ITS for hypothesis testing to include this reasoning with uncertainty.

Acknowledgments We thank teachers Jeltje Wassenberg-Severijnen and Corine Geurts for their collaboration in designing teaching tasks and delivering the course. Furthermore, we thank Noeri Huisman, Martijn Fleuren, Peter Boon and Wim van Velthoven who assisted in developing the domain module.

Compliance with Ethical Standards

Conflict of Interest none.

References

Aberson, C. L., Berger, D. E., Healy, M. R., & Romero, V. L. (2003). Evaluation of an interactive tutorial for teaching hypothesis testing concepts. *Teaching of Psychology, 30*(1), 75–78. https://doi.org/10.1207/S15328023TOP3001_12.

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *The Journal of the Learning Sciences, 4*(2), 167–207. https://doi.org/10.1207/s15327809jls0402_2.

Carver, R., Everson, M., Gabrosek, J., Horton, N., Lock, R., Mocko, M., . . . Wood, B. (2016). Guidelines for assessment and instruction in statistics education (GAISE) college report 2016. American statistical Association. http://www.amstat.org/education/gaise

Castro Sotos, A. E., Vanhoof, S., Van den Noortgate, W., & Onghena, P. (2007). Students' misconceptions of statistical inference: A review of the empirical evidence from research on statistics education. *Educational Research Review, 2*(2), 98–113. https://doi.org/10.1016/j.edurev.2007.04.001.

Drijvers, P., Boon, P., Doorman, M., Bokhove, C., & Tacoma, S. (2013). Digital design: RME principles for designing online tasks. In C. Margolinas (Ed.), *Proceedings of ICMI study 22 task Design in Mathematics Education* (pp. 55–62). Clermont-Ferrand: ICMI.

Falk, R., & Greenbaum, C. W. (1995). Significance tests die hard: The amazing persistence of a probabilistic misconception. *Theory & Psychology, 5*(1), 75–98. https://doi.org/10.1177/0959354395051004.

Field, A. (2009). *Discovering statistics using SPSS* (3rd ed.). London: Sage publications.

Garfield, J. B., Ben-Zvi, D., Chance, B., Medina, E., Roseth, C., & Zieffler, A. (2008). Learning to reason about statistical inference. *Developing students' statistical reasoning* (pp. 261–288). Dordrecht, the Netherlands: Springer.

Goguadze, G., & Melis, E. (2009). Combining evaluative and generative diagnosis in ACTIVEMATH. In V. Dimitrova, R. Mizoguchi, B. du Boulay & A. Graesser (Eds.), *Artificial Intelligence in Education* (pp. 668–670). https://doi.org/10.3233/978-1-60750-028-5-91.

Heeren, B., & Jeuring, J. (2014). Feedback services for stepwise exercises. *Science of Computer Programming, 88*, 110–129. https://doi.org/10.1016/j.scico.2014.02.021.

Hox, J. J., Moerbeek, M., & van der Schoot, R. (2018). *Multilevel analysis: Techniques and applications* (3rd ed.). New York: Routledge.

Kluger, A. N., & DeNisi, A. (1996). The effects of feedback interventions on performance. *Psychological Bulletin, 119*(2), 254–284. https://doi.org/10.1037/0033-2909.119.2.254.

Kodaganallur, V., Weitz, R. R., & Rosenthal, D. (2005). A comparison of model-tracing and constraint-based intelligent tutoring paradigms. *International Journal of Artificial Intelligence in Education, 15*, 117–144.

McCambridge, J., Witton, J., & Elbourne, D. R. (2014). Systematic review of the Hawthorne effect: New concepts are needed to study research participation effects. *Journal of Clinical Epidemiology, 67*(3), 267–277. https://doi.org/10.1016/j.jclinepi.2013.08.015.

Mitrovic, A., Koedinger, K. R., & Martin, B. (2003). A comparative analysis of cognitive tutoring and constraint-based modeling. In P. Brusilovski, A. Corbett, & F. de Rosis (Eds.), *User Modeling 2003* (pp. 313–322). Berlin Heidelberg: Springer.
Mitrovic, A., Martin, B., & Suraweera, P. (2007). Intelligent tutors for all: The constraint-based approach. *IEEE Intelligent Systems, 4*, 38–45. https://doi.org/10.1109/MIS.2007.74.
Narciss, S., Sosnovsky, S., Schnaubert, L., Andrès, E., Eichelmann, A., Goguadze, G., & Melis, E. (2014). Exploring feedback and student characteristics relevant for personalizing feedback strategies. *Computers & Education, 71*, 56–76. https://doi.org/10.1016/j.compedu.2013.09.011.
Nwana, H. S. (1990). Intelligent tutoring systems: An overview. *Artificial Intelligence Review, 4*(4), 251–277.
Roll, I., Aleven, V., & Koedinger, K. R. (2010). The invention lab: Using a hybrid of model tracing and constraint-based modeling to offer intelligent support in inquiry environments. In V. Aleven, J. Kay, & J. Mostow (Eds.), *Intelligent tutoring systems, 10th international conference* (pp. 115–124). Berlin Heidelberg: Springer-Verlag. https://doi.org/10.1007/978-3-642-13388-6_16.
Shute, V. J. (2008). Focus on formative feedback. *Review of Educational Research, 78*(1), 153–189. https://doi.org/10.3102/0034654307313795.
Sosa, G. W., Berger, D. E., Saw, A. T., & Mary, J. C. (2011). Effectiveness of computer-assisted instruction in statistics: A meta-analysis. *Review of Educational Research, 81*(1), 97–128. https://doi.org/10.3102/0034654310378174.
Steenbergen-Hu, S., & Cooper, H. (2014). A meta-analysis of the effectiveness of intelligent tutoring systems on college students' academic learning. *Journal of Educational Psychology, 106*(2), 331–347. https://doi.org/10.1037/a0034752.
Vallecillos, A. (1999). Some empirical evidences on learning difficulties about testing hypotheses. *Bulletin of the International Statistical Institute: Proceedings of the Fifty-Second Session of the International Statistical Institute, 58*, 201–204.
Van der Kleij, F., Feskens, R., & Eggen, T. (2015). Effects of feedback in a computer-based learning environment on students' learning outcomes. A meta-analysis. *Review of Educational Research, 85*(4), 475–511. https://doi.org/10.3102/0034654314564881.
VanLehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist, 46*(4), 197–221. https://doi.org/10.1080/00461520.2011.611369.
Woolf, B. P. (2009). *Building intelligent interactive tutors*. Burlington: Morgan Kaufmann Publishers.

This article includes revised parts from two short paper contributions to conferences: the Conference on European Research in Mathematics Education (2019) and the Artificial Intelligence in Education conference (2019).