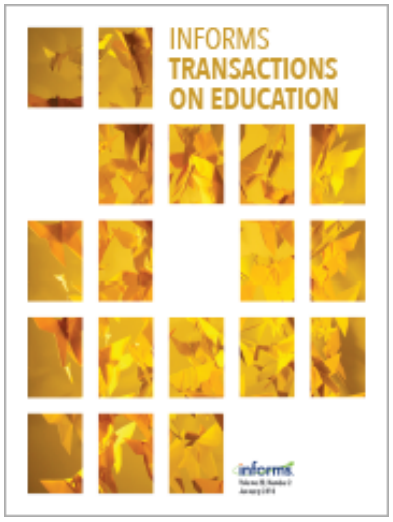
This article was downloaded by: [216.73.216.221] On: 10 November 2026, At: 15:47
Publisher: Institute for Operations Research and the Management Sciences (INFORMS)
INFORMS is located in Maryland, USA



INFORMS Transactions on Education

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Generative AI as Scaffolding for Operations Research Education: Framework, Classroom Evaluation, and Design Principles

Luis Fernando Pérez Armas, Stephania Kossman

To cite this article:
Luis Fernando Pérez Armas, Stephania Kossman (2026) Generative AI as Scaffolding for Operations Research Education: Framework, Classroom Evaluation, and Design Principles. INFORMS Transactions on Education

Published online in Articles in Advance 10 Nov 2026

. https://doi.org/10.1287/ited.2025.0167

This work is licensed under a Creative Commons Attribution 4.0 International License. You are free to copy, distribute, transmit and adapt this work, but you must attribute this work as "*INFORMS Transactions on Education*. Copyright © 2026 The Author(s). https://doi.org/10.1287/ited.2025.0167, used under a Creative Commons Attribution License: https://creativecommons.org/licenses/by/4.0/."

Copyright © 2026 The Author(s)

Please scroll down for article—it is on subsequent pages

With 12,500 members from nearly 90 countries, INFORMS is the largest international association of operations research (O.R.) and analytics professionals and students. INFORMS provides unique networking and learning opportunities for individual professionals, and organizations of all types and sizes, to better understand and use O.R. and analytics tools and methods to transform strategic visions and achieve better outcomes.
For more information on INFORMS, its publications, membership, or meetings visit http://www.informs.org

https://pubsonline.informs.org/journal/ited

INFORMS TRANSACTIONS ON EDUCATION
Articles in Advance, pp. 1–21
ISSN 1532-0545 (online)

Generative AI as Scaffolding for Operations Research Education: Framework, Classroom Evaluation, and Design Principles

Luis Fernando Pérez Armas,[a,*] **Stephania Kossman**[b]

[a] IESEG School of Management, Univ. Lille, CNRS, UMR 9221, Lille Economie Management, Lille F-59000, France; [b] Savila Education, Lille F-59800, France

*Corresponding author

l.perezarmas@ieseg.fr (L. F. Pérez Armas), hello@savilaeducation.com (S. Kossman)

https://orcid.org/0000-0002-3680-2377 (L. F. Pérez Armas), https://orcid.org/0000-0001-6188-9526 (S. Kossman)

Received: August 20, 2025
Revised: May 18, 2026
Accepted: July 3, 2026
Published Online in Articles in Advance: November 10, 2026

https://doi.org/10.1287/ited.2025.0167

Copyright: © 2026 The Author(s)

Open Access Statement: This work is licensed under a Creative Commons Attribution 4.0 International License. You are free to copy, distribute, transmit and adapt this work, but you must attribute this work as "*INFORMS Transactions on Education*. Copyright © 2026 The Author(s). https://doi.org/10.1287/ited.2025.0167, used under a Creative Commons Attribution License: https://creativecommons.org/licenses/by/4.0/."

Abstract. Operations research (OR) education requires students to progress through a demanding sequence of cognitive transitions, from problem framing and mathematical formulation to computational implementation, validation, and interpretation, yet scalable instructional support for these transitions remains limited. This article proposes a task-centered framework that maps generative artificial intelligence (GenAI) support functions to the main bottlenecks of OR learning and specifies the instructor role needed at each stage. We illustrate the framework through six use cases, organized by implementation complexity and evaluated in terms of pedagogical role and risks. We then evaluate one classroom implementation in depth: a study in which 66 master's students used a customized GPT companion to implement and solve a quadratic multiknapsack case in Python and Pyomo. Mixed-methods evidence from pre-post surveys, regression and mediation analyses, a compact path model, and qualitative coding shows that the intervention worked through manageability rather than task ease. Confidence gains were large and concentrated in the computational stages of OR modeling ($d_z = 1.29$ overall). The findings suggest that GenAI is most valuable in OR education not when it removes difficulty, but when it makes difficult modeling and implementation tasks manageable enough for students to engage, persist, and learn.

Supplemental Material: The online appendices are available at https://doi.org/10.1287/ited.2025.0167.

Keywords: operations research education • generative AI • instructional scaffolding • mathematical programming • task manageability

1. Introduction

Operations research (OR) is one of the most versatile and impactful analytical disciplines, with applications spanning logistics, healthcare, manufacturing, energy, and public policy. By transforming data and models into actionable insights, OR delivers substantial economic and societal value, enabling decisions that are more efficient, cost-effective, and evidence based than those made by intuition alone. Yet despite this potential, OR is often perceived as a niche field (Le et al. 2025). Its accessibility remains constrained by steep learning curves in mathematical modeling, optimization, and coding, as well as by limited availability of engaging and scalable teaching resources. Compared with other computational domains such as data science or machine learning, OR has not experienced the same surge in democratization, even as demand for OR professionals grows rapidly, with U.S. employment of operations research analysts projected to increase by 23% between 2023 and 2033 (Bureau of Labor Statistics 2025).

Teaching and learning OR is intrinsically challenging. Students must translate messy, real-world problems into formal models, integrate multiple objectives and constraints, and select, implement, and validate suitable solution approaches (Liebman 1998, Nestler et al. 2012). Although OR algorithms are often theoretically transparent, their practical applications can appear opaque, especially when tradeoffs are involved. Traditional OR courses, typically concentrated at the graduate level, are demanding to complete, and online offerings have struggled with low completion rates in part because of the field's complexity. For OR to meet its full potential, its education must become more inclusive, engaging, and adaptive.

Herbert Simon called for greater integration between OR and artificial intelligence (AI) nearly four decades ago, anticipating the synergies that might emerge

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

between the two fields (Simon 1987). That vision is now closer to reality. The rapid emergence of generative AI (GenAI) provides a new opportunity to expand the accessibility of OR education. Early evidence from outside OR is compelling: Large-scale field experiments show that GenAI can produce learning gains comparable to an additional year of schooling in resource-constrained contexts, at a fraction of the cost (De Simone et al. 2025). Such results must be interpreted cautiously, given risks of overreliance and cognitive shortcuts (Bastani et al. 2025, Kosmyna et al. 2025), but they suggest that even modest gains in OR education could be transformative.

Recent studies in programming, statistics, and writing underscore that the impact of GenAI depends not on its mere availability but on the design of pedagogical experiences that harness its strengths while mitigating its risks (Mollick and Mollick 2022, Ellis and Slade 2023, Bien and Mukherjee 2025, Bray 2025, Fang et al. 2025). Used well, AI tutors can support metacognition, active recall, and critical reasoning. Used poorly, they risk eroding long-term mastery. Despite this growing body of evidence, applications of GenAI in OR education remain virtually unexplored, with only a handful of preliminary efforts in related areas such as analytics education (Bray 2025).

This article addresses that gap by developing and evaluating a structured approach for integrating GenAI into OR education, with a focus on prescriptive analytics, mathematical programming, and model implementation. We argue that GenAI is most valuable when embedded in instructor-led learning designs that preserve productive struggle while making difficult modeling and implementation tasks manageable. GenAI is not treated here as a replacement for the instructor or as an answer generator, but as a scaffold that supports students as they move from problem understanding to mathematical formulation, implementation, debugging, solution extraction, and interpretation.

The paper makes four contributions. First, we propose a conceptual framework that maps the cognitive demands of OR learning to specific GenAI affordances, including natural-language explanation, Socratic prompting, code assistance, feedback, and guided reflection. Second, we synthesize six GenAI use cases for OR education, ranging from lightweight supports such as audio-based learning and flashcards to more advanced tools such as customized tutors, retrieval-augmented systems, fine-tuned models, and agentic architectures, and compare them in terms of pedagogical role, implementation complexity, required resources, and risks. Third, we implement and evaluate one use case in depth: A classroom study in which students use a customized GPT tutor to support the implementation of a quadratic multiknapsack variant implemented in Python-Pyomo and solved by HiGHS, embedded in a two-session instructional protocol combining instructor scaffolding, a preclass podcast, a case study, a validation interface, and instructor-led debriefing. Fourth, we provide mixed-methods evidence from student surveys, self-reported performance indicators, regression models, exploratory mediation analysis, a compact path model, and qualitative responses. The central finding is that the intervention's value lies not in making the task easy but in making a difficult OR implementation task manageable: Perceived GPT usefulness and integrated instructor-GPT support are associated with task manageability, manageability is associated with perceived learning, and perceived learning is associated with confidence gains and satisfaction.

All implementation material accompanying this article can be downloaded from the GitHub Repository (https://github.com/ceche1212/GenAI_for_OR_education).[1]

The remainder of this article proceeds as follows. Section 2 reviews the literature on OR education, cognitive bottlenecks in mathematical modeling, and GenAI-supported learning, and introduces the conceptual framework. Section 3 synthesizes the proposed use cases and presents an adoption roadmap for OR instructors. Section 4 describes the classroom intervention, measures, and analysis strategy. Section 5 reports the quantitative and qualitative findings. Section 6 discusses design implications, limitations, and future research.

2. Conceptual Framework: GenAI as Scaffolding for or Learning

This section develops the conceptual basis for integrating GenAI into OR education. Rather than reviewing GenAI architectures in technical detail, the focus is on the pedagogical problem that motivates the paper: Students must move through a sequence of cognitively demanding OR tasks, from understanding a messy decision situation to formulating, implementing, validating, and communicating a model-based recommendation. We first summarize the main bottlenecks in OR learning and the instructional supports proposed in the literature. We then discuss the affordances and risks of GenAI for these bottlenecks and present a framework that maps specific GenAI supports to OR learning tasks and instructor roles.

2.1. Cognitive Bottlenecks in OR Education

The teaching of OR and mathematical modeling has progressively moved from lecture-based delivery toward more active, situated, and student-centered pedagogies. This shift is natural because OR is an applied discipline: Students must learn not only algorithms, but also how to frame decision problems, structure assumptions, formulate models, implement them, validate results, and communicate insights in realistic contexts (Brandon-Jones et al. 2012, Drake 2019). Case-based

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

learning, simulations, thematic assignments, games, and digital learning environments have all been used to make OR learning more experiential and engaging (Grinde and Kammermeyer 2003, DePuy and Taylor 2007, Dias 2017, Frances and Terekhov 2019, Joseph-Richard and Uhomoibhi 2024, Beliën et al. 2025, Van den Broeke and Mlinar 2025). However, these methods are effective only when they preserve the core cognitive work of OR rather than replacing it with surface-level activity.

A recurring theme in the OR education literature is that the skills required for professional practice extend well beyond computational proficiency. Students must frame and scope problems, prepare and interpret data, select appropriate analytical approaches, construct valid and adaptable models, deploy solutions, and translate outputs into managerial insights (Nestler et al. 2012, Drake 2019). From a cognitive perspective, these tasks require combining declarative, procedural, and conditional knowledge (Liebman 1998). From a modeling perspective, OR models function not only as computational artifacts but also as maps or "microworlds" that help learners reason about complex systems and refine their mental models (Morecroft 1992).

The difficulty of OR learning therefore lies in the transitions between stages. Students may understand a problem narrative but fail to define appropriate decision variables. They may understand a mathematical formulation but struggle to translate it into a modeling language such as Python with Pyomo or into a spreadsheet model. They may obtain solver output but fail to validate, interpret, or communicate it. These transitions are especially demanding because they require abstraction, schema development, technical implementation, and critical interpretation simultaneously. Effective OR instruction must therefore provide scaffolding at the points where students are most likely to become stuck, while preserving the productive struggle needed for durable learning.

Figure 1 summarizes this workflow. Although the stages are shown sequentially, OR learning is inherently iterative: Implementation may reveal a formulation error, validation may require revisiting assumptions, and interpretation may expose the need for additional analysis. Table 1 summarizes common student bottlenecks at each stage and examples of instructional supports that reduce cognitive load without eliminating the core reasoning task.

Three design implications follow from this literature. First, OR learning benefits from active and situated tasks, such as cases, projects, and realistic data sets, because these require students to move repeatedly through the stages of the OR workflow. Second, scaffolds and feedback systems are most valuable when they target specific bottlenecks, including formulation, method selection, implementation, debugging, and interpretation (Liebman 1998, Reisman 2004, Scheubrein and Kulturel-Konak 2006, Dean 2020). Third, motivational mechanisms must be integrated with authentic learning goals: Games, simulations, and digital supports can increase engagement, but poorly aligned gamification or excessive automation can reduce learning if students bypass the reasoning processes that OR instruction is meant to develop (Hammedi et al. 2021, Kwon and Özpolat 2021). These implications motivate the framework developed below. GenAI should be evaluated not by whether it gives students faster answers, but by whether it provides targeted scaffolding that makes difficult OR tasks manageable while preserving student reasoning and instructor guidance.

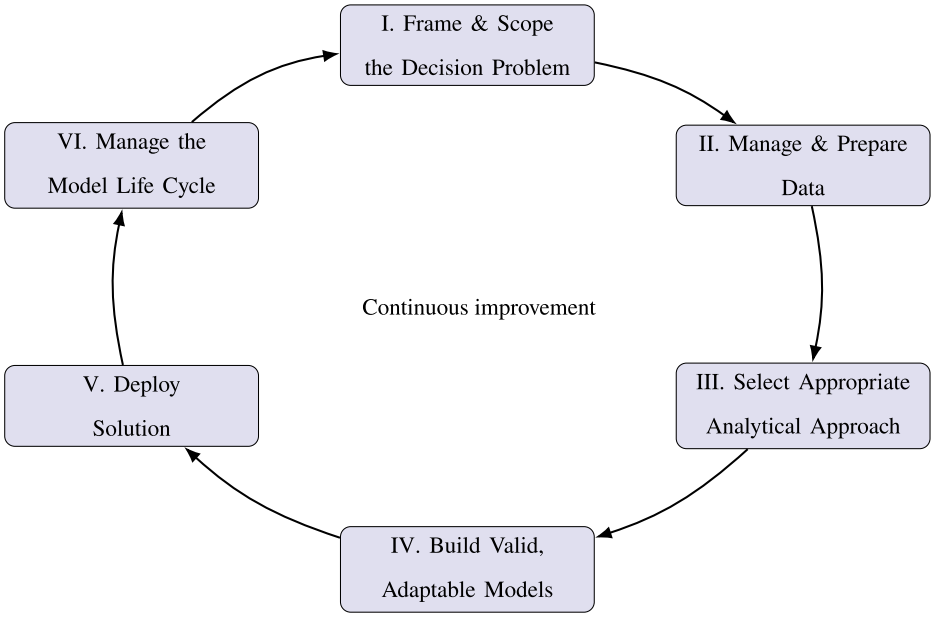
2.2. GenAI Affordances for or Learning

AI has historically entered OR education through several distinct routes: as curricular content, with machine learning and optimization introduced within OR and analytics programs (Boutilier and Chan 2023); as a logistics tool, with AI algorithms used to improve scheduling, resource allocation, and student-tutor matching (Xu 2025); as expert systems supporting students in specific modeling tasks through step-by-step feedback (Scheubrein and Kulturel-Konak 2006); and as solution methods, with reinforcement learning and related approaches studied as solvers for OR problems (Sutton et al. 1999, Schmidhuber 2015, Bertsekas 2025). These applications reflect the long-standing methodological connection between AI and OR (Simon 1987, Boyd and Vandenberghe 2004, Bennett and Parrado-Hernández 2006, Van Hentenryck and Bent 2006). However, none of them fully addresses the pedagogical bottlenecks identified in Section 2.1, particularly the transitions from problem understanding to formulation, from formulation to code, and from solver output to interpretation.

Generative AI changes this opportunity space because it can interact with students through natural language, code, mathematical notation, examples, and iterative feedback. In education more broadly, GenAI tools have provided on-demand explanations, generated localized learning materials, and supported problem solving in resource-constrained settings (De Simone et al. 2025). Pedagogical research suggests that GenAI can strengthen transfer, metacognition, and critique-based learning when students are asked to evaluate or challenge AI-generated explanations rather than merely consume them (Mollick and Mollick 2022). In programming and data analytics education, GenAI has been used to reduce the syntax barrier, support experimentation, and enable students to focus on computational logic and interpretation (Ellis and Slade 2023, Bien and Mukherjee 2025, Bray 2025). Multimodal and retrieval-based systems further extend these

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

Figure 1. OR Learning Workflow with Iterative Feedback and Continuous Improvement



possibilities by enabling richer, more context-specific feedback (Fang et al. 2025).

For OR education, the most relevant affordance of GenAI is not autonomous problem solving. Current generative models should not be treated as substitutes for specialized optimization algorithms or solvers, especially in complex reasoning tasks (Shojaee et al. 2025). Rather, their educational value lies in supporting the surrounding cognitive work: helping students unpack a problem statement, clarify assumptions, identify modeling components, translate mathematical expressions into executable code, debug implementation errors, interpret solver output, and reflect on the validity of a recommendation. This is particularly relevant for mathematical programming courses, where students often understand isolated concepts but struggle to coordinate notation, data structures, code, solver behavior, and managerial interpretation.

These affordances can be grouped into six categories. First, GenAI can provide *interactive explanation*, allowing

Table 1. Cognitive Bottlenecks in OR Learning and Examples of Instructional Supports

OR learning stage	Common student bottlenecks	Instructional supports
Framing and scoping the decision problem	Misidentifying decision makers, objectives, constraints, or relevant assumptions; treating an ambiguous context as a fully specified exercise	Realistic cases, guided questioning, peer discussion, and assumption-checking activities (Drake 2019, Frances and Terekhov 2019)
Managing and preparing data	Overlooking data quality problems; misunderstanding parameter definitions; organizing data in ways that do not match the model structure	Hands-on work with messy datasets, data-cleaning exercises, and documentation of assumptions
Selecting an analytical approach	Choosing methods by familiarity rather than fit; overlooking problem structure, data limitations, interpretability, or implementation constraints	Method taxonomies, decision trees, and discussion of tradeoffs across accuracy, interpretability, and time (Goffin 1998, Reisman 2004)
Building valid and adaptable models	Misdefining decision variables; writing incomplete objectives; omitting or misrepresenting constraints; failing to connect notation to the problem narrative	Worked examples, think-aloud modeling, model critique, and repeated translation between natural language and algebra (Liebman 1998)
Deploying and validating solutions	Incorrect coding or solver setup; indexing errors; weak debugging; limited testing for infeasibility, unboundedness, or inconsistent outputs	Electronic teaching assistants, debugging checklists, structured laboratory sessions, and learning assistants (Scheubrein and Kulturel-Konak 2006, Dean 2020)
Interpreting and communicating results	Treating solver output as self-explanatory; failing to explain tradeoffs, assumptions, limitations, and managerial implications	Case debriefs, sensitivity analysis, presentation tasks, and stakeholder-oriented reporting (Drake 2019, Frances and Terekhov 2019)

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

students to ask follow-up questions about concepts, notation, constraints, or code. Second, it can support *Socratic prompting*, guiding students through decisions without immediately providing complete answers. Third, it can offer *code assistance*, including template generation, syntax explanation, and translation between mathematical formulations and modeling languages such as Pyomo, Gurobi, or CPLEX. Fourth, it can support *debugging and troubleshooting*, helping students diagnose syntax, indexing, and solver errors. Fifth, it can enable *personalized guidance*, adapting explanations to students' prior knowledge and allowing repeated practice outside instructor contact hours. Sixth, it can support *reflection and critique*, for example by asking students to verify whether a generated formulation matches the original decision problem.

At the same time, these affordances create risks that are especially salient in OR. GenAI systems can produce plausible but incorrect formulations, code, or explanations (Cribben and Zeinali 2023). Students may overrely on generated answers, bypassing the abstraction and validation work that OR education is meant to develop (Bastani et al. 2025, Kosmyna et al. 2025). These risks are amplified in quantitative domains because small errors in indices, constraints, or parameter definitions can produce solutions that appear reasonable but are mathematically invalid. GenAI-supported OR learning therefore requires explicit verification practices, instructor-designed guardrails, and debriefing activities that help students distinguish between syntactic help, modeling reasoning, and valid optimization logic.

The framework developed in this paper treats GenAI as a support structure rather than an answer engine. Its role is to make difficult learning tasks more manageable while preserving student reasoning and instructor oversight. Because the GenAI ecosystem is evolving rapidly, the focus here is not on an exhaustive technical account of specific architectures, but on identifying how different forms of GenAI support can assist students at particular points in the OR learning process. The use cases discussed in this paper should therefore be read as pedagogical design patterns rather than fixed technological prescriptions. Additional background on more advanced GenAI architectures is provided in Online Appendix B for readers interested in the technical foundations of the higher-complexity use cases discussed later in this article.

2.3. Mapping GenAI Supports to or Learning Tasks

Building on the workflow in Figure 1 and the bottlenecks in Table 1, we define a task-centered framework for GenAI-supported OR learning. The framework starts from the cognitive work students must perform rather than from a list of AI tools. For each OR learning stage, the relevant question is not simply whether GenAI can produce an answer, but whether it can provide targeted support at the point where students are likely to become stuck. This distinction is central for OR education, where a syntactically correct formulation, code fragment, or explanation can still be pedagogically harmful if it bypasses the reasoning and validation work students need to develop.

The framework has four dimensions: the OR learning task, the student bottleneck, the GenAI support function, and the instructor role. Table 2 presents the resulting mapping.

A further implication of the framework is that GenAI support can help bridge the gap between canonical textbook problems and more realistic OR applications. Introductory courses often focus on basic problem families because they are essential for teaching modeling patterns, but this can make OR appear abstract or disconnected from practice. The knapsack problem, for example, is valuable for introducing binary decisions and capacity constraints, yet real applications rarely stop at the single-constraint version: Portfolio selection leads toward quadratic or multiknapsack variants, logistics problems lead toward bin packing or container loading, and production settings often require additional precedence or resource constraints. GenAI can make these extensions more accessible by helping students understand how a familiar modeling pattern changes when the context becomes more realistic, supporting a progression from archetypal models to richer applied formulations.

Three design principles follow from this framework. First, GenAI support should be targeted to specific bottlenecks rather than deployed as general-purpose answer generation. Second, instructor guidance must remain central at moments of abstraction, validation, and consolidation, because these are precisely the points where plausible AI output can conceal conceptual errors. Third, the aim should be manageability rather than easiness: well-designed GenAI support helps students remain engaged with difficult modeling and implementation tasks without removing the difficulty that makes OR learning meaningful. The next section translates this framework into a set of concrete use cases and an adoption roadmap for OR instructors.

3. GenAI Use Cases and Adoption Roadmap for OR Education

The framework in Section 2 argues that GenAI should be integrated into OR education as targeted support for specific learning bottlenecks. This section translates that framework into a set of pedagogical use cases, organized by increasing implementation complexity. The goal is not to present a fixed catalogue of tools, because GenAI systems evolve rapidly, but to identify design patterns that OR instructors can adapt across

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

Table 2. Framework Mapping GenAI Support Functions to OR Learning Stages and Instructor Roles

OR learning stage	Student bottleneck	GenAI support function	Instructor role
Framing and scoping the decision problem	Students may read the problem superficially, miss stakeholders, overlook objectives or constraints, or treat ambiguity as if it were already resolved.	Reframe the context in accessible language, generate guiding questions, surface assumptions, and provide alternative descriptions of the decision situation.	Curate realistic contexts, define learning objectives, challenge weak assumptions, and ensure that students do not confuse narrative fluency with valid problem framing (Drake 2019, Frances and Terekhov 2019).
Managing and preparing data	Students may misunderstand parameter definitions, organize data in structures that do not match the formulation, or overlook missing, inconsistent, or irrelevant information.	Explain parameter meanings, suggest data organization strategies, generate small illustrative examples, and help students connect data structures to sets, indices, and parameters.	Validate data assumptions, emphasize documentation, and ensure that data preparation decisions remain aligned with the modeling purpose.
Selecting an analytical approach	Students may choose a method because it is familiar rather than because it fits the problem structure, available data, computational constraints, or interpretation needs.	Compare candidate modeling approaches, explain tradeoffs, retrieve analogies to related problem classes, and prompt students to justify methodological choices.	Guide method selection, correct oversimplified recommendations, and connect choices to OR taxonomies and professional judgment (Goffin 1998, Reisman 2004).
Building valid and adaptable models	Students may struggle to translate natural language into decision variables, objective functions, and constraints, or fail to connect mathematical notation to the problem narrative.	Use Socratic questioning, step-by-step explanation, formulation critique, and alternative representations to help students construct and revise the model.	Preserve productive struggle, test the logic of the formulation, and help students distinguish mathematically valid models from plausible but incorrect ones (Liebman 1998, Mollick and Mollick 2022).
Deploying and validating solutions	Students may struggle to translate the formulation into a computational environment and may face indexing errors, solver messages, infeasibility, or weak validation practices.	Explain code or model syntax, generate pseudocode, suggest implementation templates, interpret error messages, support debugging, and propose validation checks.	Verify alignment between the mathematical model and its implementation, distinguish coding errors from modeling errors, and reinforce testing and validation practices (Scheubrein and Kulturel-Konak 2006, Dean 2020, Bien and Mukherjee 2025).
Interpreting and communicating results	Students may treat solver output as the final answer, overlook assumptions and tradeoffs, or struggle to translate numerical results into managerial implications.	Explain solution outputs, help structure sensitivity questions, draft alternative explanations, and support reflection on limitations and recommendations.	Validate the interpretation, connect results to stakeholder needs, emphasize limitations, and lead debriefs that consolidate conceptual learning (Drake 2019, Frances and Terekhov 2019).
Retention and transfer across problem families	Students may forget modeling patterns, fail to recognize related structures in new problems, or memorize examples without abstracting transferable principles.	Generate retrieval-practice prompts, varied examples, self-quizzes, and comparisons across problem families to support active recall and transfer.	Curate core modeling patterns, sequence practice over time, and help students connect specific cases to general modeling principles (Liebman 1998, Mollick and Mollick 2022).

courses, modeling environments, solvers, and institutional contexts.

3.1. Overview of the Use Cases

Table 3 summarizes the six use-case families discussed in this section. The table serves as an adoption guide rather than a ranking: Low-complexity tools are most valuable when the objective is to improve engagement, preparation, or retention; medium-complexity tools are better suited for supporting students through open-ended modeling and implementation tasks; and higher-complexity tools, though more demanding to

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

Table 3. Use-Case Families for GenAI-Supported OR Education

Use-case family	OR learning stages supported	Pedagogical role	Implementation complexity	Main risks or limitations
Audio-based contextualization	Framing and scoping; approach awareness	Preclass preparation, motivation, and accessible introduction to problem contexts	Low	Passive consumption if not connected to in-class tasks
Flashcards and retrieval practice	Retention and transfer across problem families; model-building patterns	Active recall of constraints, formulations, taxonomies, and modeling techniques	Low	Shallow memorization if not linked to modeling practice
Customized tutors and case-study companions	Formulation; implementation; debugging; interpretation	Guided problem solving, Socratic questioning, and individualized support	Medium	Overreliance, plausible but incorrect guidance, and insufficient verification
Retrieval-augmented knowledge assistants	Approach selection; formulation; implementation; validation	Grounded access to textbooks, documentation, code examples, and domain repositories	Medium-high	Retrieval quality, source coverage, maintenance, and deployment requirements
Fine-tuned or adapted models	Problem generation; structured formulation; assessment design	Consistent production of structured OR materials, such as problem instances and formulations	High	Training data requirements, cost, model drift, and maintenance
Agentic OR learning environments	Full OR workflow, from problem framing to reporting	Coordination of specialized agents for formulation, method selection, implementation, solving, validation, and communication	Very high	Reliability, governance, cost, interoperability, and risk of excessive automation

deploy, become important for scaling domain-specific knowledge bases, solver documentation support, or structured problem generation.

As implementation complexity increases, so does the need for instructional design, validation, and governance. A podcast or flashcard deck requires relatively light instructor oversight, provided it is aligned with course objectives. A customized tutor or case-study companion requires more careful prompting, guardrails, and debriefing, because it interacts directly with students during reasoning and implementation. Retrieval-augmented, fine-tuned, or agentic systems introduce additional questions of source quality, reliability, maintenance, and accountability. The following subsections discuss each use-case family in terms of how it supports the framework rather than focusing on the underlying technology.

3.2. Low-Complexity Supports: Contextualization and Retention

The first family of use cases involves low-complexity supports that instructors can adopt without specialized software infrastructure. Two examples are particularly relevant for OR education: audio-based contextualization, which helps students enter a problem domain before class, and AI-generated flashcards, which support retention of modeling patterns and technical vocabulary. In terms of the framework in Table 2, these tools primarily support framing and scoping, approach awareness, retention, and transfer across problem families.

3.2.1. Audio-Based Learning for Problem Awareness and Contextualization.

OR materials are often conceptually dense and technically specialized, making them difficult to approach without prior context. This is especially problematic at the framing and scoping stage, where students must understand the decision setting before they can formulate a model. Recent concerns about reduced reading endurance in higher education suggest that some students struggle to persist with long technical texts (Horowitch 2024), and broader evidence points to declining reading motivation and increasing consumption of audio and video media among younger students (National Endowment for the Arts 2022, National Center for Education Statistics 2023). Students who do not engage with the problem context risk treating modeling as a mechanical exercise rather than as a structured response to a decision problem.

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

Audio-based learning can address this entry point. Research on audiobooks suggests that listening supports comprehension for struggling readers, particularly when combined with print materials (Singh and Alexander 2022), and studies of podcasting in higher education report benefits for comprehension, retention, and student engagement (Gunderson and Cumming 2023). In OR courses, instructor-curated audio can introduce the business or operational context of a problem, explain why it matters, and connect it to a recognizable modeling family before formal notation is introduced, consistent with the taxonomic approach to OR education advocated by Reisman (2004).

GenAI lowers the cost of producing such materials. Text-to-speech and audio-overview tools can transform case descriptions, literature reviews, or methodological summaries into short podcast-style introductions. For example, an instructor may convert a reading on order batching into a conversational audio segment explaining why warehouse travel time matters, how batching affects operations, and why the problem belongs to a broader family of routing and grouping decisions (Pardo et al. 2024). The pedagogical objective is not to replace reading or modeling, but to prepare students to engage with the case and connect the narrative to the OR workflow. A sample podcast excerpt and implementation details are provided in Online Appendix A.

The main risk is passive consumption. If audio content is treated as entertainment rather than preparation, it may not improve learning. The instructor's role is therefore to connect the material to an active preclass task, such as asking students to identify the decision maker, possible objectives, constraints, or related OR problem families. Used this way, audio-based learning functions as a low-cost tool for contextualization and problem awareness.

3.2.2. Spaced Repetition and Active Recall for Modeling Patterns. OR modeling requires students to develop fluency with recurring structures: assignment constraints, capacity constraints, flow-balance constraints, precedence constraints, logical constraints, and linearization techniques. Without this repertoire, students struggle to recognize familiar structures in new problem contexts and to progress toward approach selection or computational implementation. This is one reason why canonical problems remain pedagogically important: students learn transferable patterns from simpler models before adapting them to richer applications.

This type of structured knowledge is however vulnerable to forgetting. The forgetting curve shows that recently learned information decays rapidly without repeated retrieval (Ebbinghaus 1913). Active recall, meaning deliberate retrieval from memory rather than passive review, strengthens memory and supports durable learning (Szilard 1964). When combined with spaced repetition, retrieval practice improves long-term retention and transfer across domains (Smolen et al. 2016, Smith and Scarf 2017, Latimier et al. 2021). Digital flashcard platforms such as *Anki* or *Quizlet* operationalize these principles by scheduling difficult items more frequently and easy items less so, and have been associated with self-regulatory benefits including goal setting and progress monitoring (Boroughani et al. 2023).

In OR education, flashcards can support both factual recall and structural recognition: One card might ask students to recall how to model mutual exclusivity between binary variables, another to identify the role of a flow-balance constraint, and another to recognize when a big-M formulation is needed. These cards do not replace modeling practice but help students build a mental library of reusable modeling components, which is essential for transfer because realistic OR problems typically combine several familiar structures rather than introducing entirely new logic.

GenAI can reduce the instructor workload associated with creating these materials. Given lecture notes, textbook excerpts, or a list of modeling techniques, an LLM can generate draft flashcards in formats compatible with common platforms. The instructor's role remains essential: generated cards must be checked for correctness, aligned with course notation, and linked to modeling exercises, because poorly designed cards may encourage shallow memorization if students learn isolated formulas without understanding when and why they apply. Example flashcards and prompt templates for basic and cloze-style cards are provided in Online Appendix A.

3.3. Medium-Complexity Supports: AI Tutors and Case-Study Companions

The second family of use cases involves customized AI tutors and case-study companions. These tools are more demanding than audio or flashcard supports because they interact with students during reasoning, formulation, implementation, and debugging, but they are also closer to the core bottlenecks of OR education. Active learning consistently produces stronger outcomes than passive instruction (Deslauriers et al. 2011), yet active coaching is difficult to scale in OR courses, where students most need individualized feedback precisely when translating a problem into a formulation, converting a formulation into code, or diagnosing unexpected solver behavior. Customized AI tutors create an opportunity to extend this kind of guided interaction beyond what is feasible in large or time-constrained courses.

3.3.1. Socratic Tutoring and Guided Problem Solving. A Socratic AI tutor is designed not to provide complete solutions immediately, but to ask guiding

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

questions, request justification, surface missing assumptions, and help students reason through the next step (Mollick and Mollick 2022, Bray 2025). In OR, this is useful when students must identify decision variables, distinguish parameters from variables, decide whether a constraint is needed, or justify why a particular objective function matches the decision problem.

The value of this design is that it preserves productive struggle. A tutor that outputs a complete formulation may help students finish an assignment while bypassing the reasoning process. One that asks students to articulate the decision, justify the constraint logic, or compare alternative formulations instead supports metacognition and deeper understanding. This is particularly important in mathematical programming, where a model can look formally correct while still misrepresenting the actual decision problem. The instructor's role is to design the tutor's behavior, provide course-specific context, define what kinds of help are permitted, and debrief the resulting work in class.

Customized tutors are also well suited to flipped classroom settings, where students interact with the tutor before class to prepare questions or attempt a first formulation, and class time focuses on comparison, critique, and instructor feedback. This use case is not tied to a specific software ecosystem and can support spreadsheet modeling, algebraic modeling languages, or programming-based environments. Prompt templates for creating Socratic OR tutors are provided in Online Appendix A.

3.3.2. Adversarial Dialogue and Reflective Critique. A related design is the adversarial dialogue agent, sometimes described as an AI "boxer" (Bray 2025). Where a Socratic tutor guides students gently, an adversarial agent challenges their reasoning more directly: asking them to defend an assumption, explain why a constraint is valid, identify a counterexample, or justify a methodological choice. This design draws on evidence that articulating and defending reasoning is itself generative for learning, even when students initially lack the correct answer (Smith et al. 2009).

In OR education, adversarial dialogue is particularly useful for formulation critique and interpretation. Students may be asked whether a capacity constraint is sufficient, whether a decision variable is defined at the correct level of granularity, or whether a recommendation would hold under different assumptions. The pedagogical aim is reflective refinement rather than confrontation. The instructor must calibrate the level of challenge carefully, since excessive pressure may discourage weaker students while insufficient challenge fails to expose misconceptions. Prompt templates for adversarial OR dialogue agents are provided in Online Appendix A.

3.3.3. Case Study Companions. Case study companions combine the two previous roles. Case-based learning is especially valuable in OR because it exposes students to ambiguous, realistic situations requiring problem framing, model construction, and communication (Drake 2019, Frances and Terekhov 2019). However, it is also coaching intensive: Students may focus prematurely on finding "the answer" rather than exploring the modeling choices, assumptions, and tradeoffs that make the case educationally valuable. A case study companion addresses this by guiding students through an instructor-designed trajectory while requiring them to make and justify decisions at each step.

Such a companion can help students understand the setting, identify stakeholders and decisions, propose variables and constraints, challenge weak assumptions, provide hints during implementation, and support interpretation of results. It can also help students move from a canonical model to a more realistic variant by asking how the formulation changes when additional constraints or objectives are introduced, supporting both guided practice and transfer to richer problem contexts.

The main risk is that the companion becomes an answer generator. To avoid this, its instructions should specify when it may provide hints, when it should ask questions, and when it should withhold a complete solution. It should also encourage verification, for instance, by asking students to compare generated code against the original problem statement. The instructor remains responsible for defining the case trajectory, validating the final model, and leading the debrief.

As a proof of concept, we created a case study companion for the case *Resource Planning at L'Audobarquoise of Saint-Omer: A Graph Theory and Mathematical Programming Case Study with Python* (Pérez Armas 2024), designed as a boxer-coach that asks students to explain their reasoning, challenges weak answers, and guides them through the case without providing the full solution upfront. A sample interaction and the companion prompt are provided in Online Appendix A.

The classroom evaluation in Section 4 focuses on this family of tools because it targets several central OR bottlenecks simultaneously: understanding the case, formulating the model, translating it into a computational implementation, debugging, extracting the solution, and interpreting the result. This makes case study companions a particularly promising entry point in the adoption roadmap, more powerful than low-complexity preparation tools but less technically demanding than retrieval-augmented, fine-tuned, or agentic systems.

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

3.4. Higher-Complexity Supports: Knowledge Grounding, Structured Outputs, and Agentic Workflows

The higher-complexity use cases address bottlenecks that simpler tools cannot easily solve: access to specialized OR knowledge, consistency in structured content generation, and coordination across the full modeling workflow. They require more infrastructure and governance and are best viewed as advanced options for instructors or programs with a clear pedagogical objective. Technical background on the underlying architecture is provided in Online Appendix B.

3.4.1. Retrieval-Augmented Knowledge Assistants. A recurring difficulty in OR courses is that useful knowledge is fragmented across textbooks, solver manuals, modeling-language documentation, and specialized domain references. Students may not know where to search, how to interpret technical documentation, or how to connect a new problem with related formulations. Retrieval-augmented assistants address this by grounding GenAI responses in a curated knowledge base. An assistant built on mathematical programming texts, for instance, could help students compare a natural-language problem with related formulations (Williams 2013, Sánchez 2022); a project-scheduling assistant could help navigate variants of the resource-constrained project scheduling problem (Demeulemeester and Herroelen 2002; Artigues et al. 2008; Vanhoucke et al. 2012; Schwindt and Zimmermann 2015a, b); and a solver-documentation assistant could help students interpret syntax, error messages, and implementation examples across modeling environments. In all cases, the instructor's role is to curate reliable sources and teach students to verify whether retrieved explanations fit the problem at hand. Implementation material is provided in Online Appendix A.

3.4.2. Adapted Models for Structured or Content. OR instructors often need to generate multiple versions of a case or problem instance while preserving coherence among the narrative, sets, parameters, variables, objective function, and constraints. General-purpose LLMs can help draft such material, but their outputs may vary in notation, structure, or modeling assumptions. Fine-tuned or otherwise adapted models can reproduce a desired structure more reliably, for example, by generating instances in a fixed template covering problem family, story, sets, parameters, variables, objective, constraints, and notes. This can reduce preparation time and support scalable assessment design. The key limitation is that consistency is not correctness: A generated formulation may follow the template and still contain subtle modeling errors. Adapted models should therefore be treated as instructor productivity tools rather than autonomous sources of valid OR content. A structured-output example is provided in Online Appendix A.

3.4.3. Agentic or Learning Environments. OR projects rarely involve a single isolated step: They require problem framing, formulation, method selection, implementation, solving, validation, interpretation, and communication. Agentic systems could eventually coordinate specialized support for each of these stages, with one agent clarifying the decision problem, another critiquing the formulation, another assisting with implementation, and another supporting managerial reporting. Recent demonstrations show agentic systems already interacting with optimization solvers through autonomous tool calls.[2] The educational risk is that students become observers of an automated workflow rather than active modelers. Agentic systems should therefore currently be treated as a future design horizon requiring strong instructor control, transparency, and validation. A schematic illustration is provided in Online Appendix A.

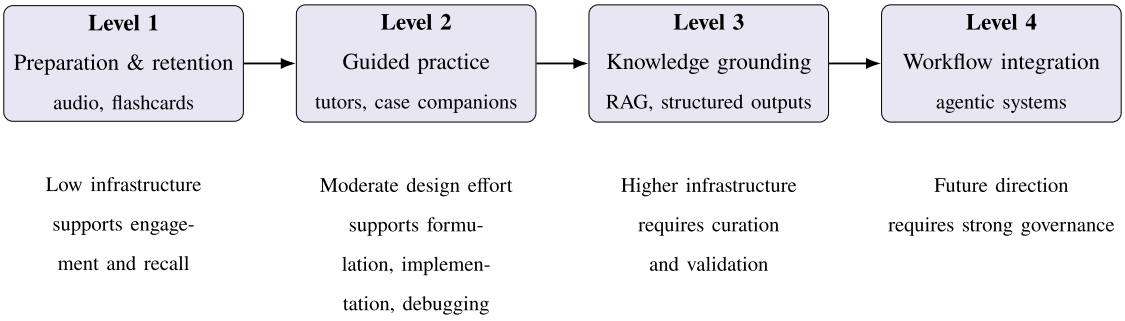
Across these three cases, the key issue is not technical sophistication but instructional alignment. Retrieval-augmented assistants are useful when students need grounded access to specialized knowledge; adapted models are useful when instructors need consistent structured outputs; and agentic systems represent a longer-term direction for integrating support across the complete OR workflow. None of these tools eliminates the need for instructor judgment; their value depends on source curation, validation, and clear boundaries on what students are expected to reason through themselves.

3.5. Adoption Roadmap

The use cases above suggest a staged adoption roadmap for OR instructors, summarized in Figure 2. The roadmap is pedagogical rather than purely technical.

Selecting the appropriate level requires instructors to first diagnose the most prominent bottlenecks their students face, assess students' prior experience with OR modeling and programming, and consider the course format. Synchronous in-person settings, asynchronous online courses, and hybrid formats each create different constraints on how GenAI support can be delivered. A further practical distinction separates the levels along the dimension of continuous model access. Low-complexity tools at Level 1 do not require students to interact with a live GenAI system: Content such as podcasts or flashcard decks is generated by the instructor using a large language model (LLM) and then consumed independently by students. From Level 2 onward, however, students must have continuous access to a generative model during the learning activity, because the educational value depends on real-time dialogue between the student and the AI.

Figure 2. Staged Adoption Roadmap for GenAI-Supported OR Education



This distinction matters for equity and institutional readiness. Not all students have reliable access to commercial LLM platforms, and not all institutions can guarantee it. Instructors should therefore consider access constraints before adopting medium- or high-complexity tools, and where possible provide shared or institution-sponsored access to avoid creating unequal learning conditions.

The classroom evaluation reported in this article illustrates a case where the dominant bottleneck was implementation. The instructor wanted students to formulate and solve a realistic OR case study involving a quadratic multiknapsack structure, an inherently demanding computational task. Expecting an instructor to debug the code of more than forty students in real time during a two-hour session is unrealistic. The alternative, showing students a complete working solution, is pedagogically passive and, even if executed flawlessly, does not guarantee learning, because students may develop the impression of understanding without having engaged in the reasoning themselves (Deslauriers et al. 2011). A GenAI companion addresses precisely this gap: It gives every student access to individualized, real-time implementation support, functioning as a dedicated teaching assistant for each learner during the session. The tools evaluated in this study therefore sit at the frontier of Levels 2 and 3, combining guided interaction with instructor-curated knowledge to support problem understanding, formulation, implementation, debugging, and interpretation. The following sections evaluate whether this instructor-led GenAI workflow can make a challenging OR implementation task manageable and support learning, confidence, and satisfaction.

4. Classroom Intervention and Evaluation Design

We evaluated the framework in Table 2 through a classroom OR modeling activity targeting the bottlenecks most difficult to address in a single session: problem understanding, formulation, computational implementation, solution extraction, and interpretation of a realistic and technically demanding OR problem. The GenAI tool was implemented as an instructor-designed case companion, as described in Section 3.3.3, intended to support the task without replacing student reasoning or instructor debriefing. The evaluation combines pre- and postsurveys, self-reported performance indicators, and open-ended responses.

4.1. Educational Context and Participants

The intervention was conducted within an Agile Project Management course offered in the second semester of the first year of a master's program at a French Business School. The relevant sessions focused on agile project planning from the perspective of the product owner. Students were enrolled in the Supply Chain and Operations Management track and had completed a decision modeling course in the previous semester covering basic linear and integer programming with Excel Solver, NEOS Server, and CPLEX. Most had minimal Python experience, and for many, this was their first exposure to algebraic modeling through Python-based tools.

The intervention was implemented across two in-person class groups: 46 students on the Lille campus and 27 on the Paris campus, for a total of 73 enrolled students. Students worked individually on the case across two two-hour sessions. Participation in the pre- and postsurveys was voluntary, based on informed research consent, and the data were anonymized before analysis. The final matched sample includes 66 students who completed both surveys and participated in the full activity sequence.

4.2. Learning Objectives, Case Study, and Intervention Protocol

The intervention was built around the case study *Six Sprints to Survive: Value and Data-Driven Agile Planning at SynapseFlow*, a sprint-planning problem inspired by the quadratic multiknapsack structure studied by Boschetti et al. (2014). Students decide which software user stories[3] to develop, and in which sprint,[4] over a six-sprint planning horizon. Each story consumes effort and generates business value; complementary stories

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

yield additional interaction value; later delivery is discounted; and precedence constraints encode logical dependencies among features.

The learning objectives were to map the case narrative to a mathematical programming model, implement and solve it in Python using Pyomo[5] and HiGHS[6] in Google Colab,[7] extract and validate the solution, interpret the result, and use GenAI critically as a support tool for modeling and implementation. Figure 3 summarizes the protocol.

Session 1 introduced single-sprint planning through a knapsack model. The class derived the formulation on the whiteboard, implemented and solved a 100-item instance in Pyomo and HiGHS, extracted and interpreted the solution, and completed the presurvey. Between sessions, students received a 16-minute GenAI-generated podcast introducing the case context.

Session 2 took place one week later. The instructor introduced interactions between user stories, quadratic objective terms, and a standard linearization strategy using one auxiliary variable and three constraints. After a worked quadratic knapsack example, students read the case for 15 minutes and then worked individually for approximately one hour on the full implementation. To complete the case, students received an instructor-prepared Google Colab notebook that included predefined data-import functions and the problem data but left the model implementation blank. Students were responsible for defining the decision variables, objective function, and constraints in Pyomo, configuring the HiGHS solver, extracting the solution, and validating feasibility. They were strongly encouraged to use the AI companion throughout and could verify feasibility through a simple validation interface provided in the notebook. During the implementation phase, the instructor answered only access-related questions and redirected all modeling and programming questions to the companion.

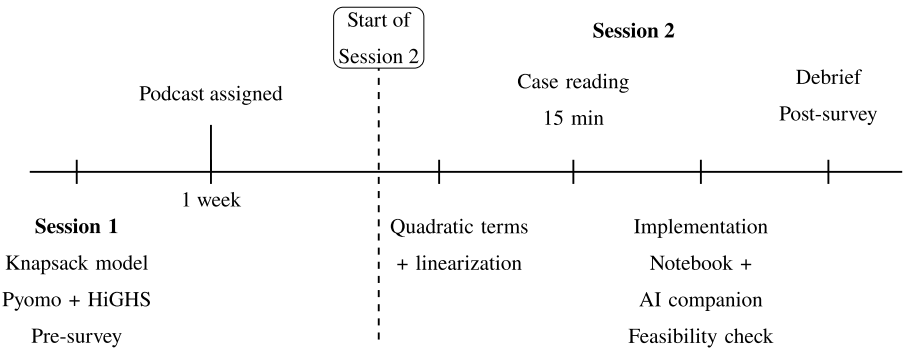
The session closed with an instructor-led debrief covering common implementation roadblocks, constraint construction, solution extraction, feasibility checking, interpretation of sprint assignments, and sensitivity analysis on the discount rate. Students then completed the postsurvey, administered through Microsoft Forms with institutional accounts restricted to one response per student. Performance indicators were self-reported, including whether students solved the case, their best objective value, and the number of attempts.

4.3. GenAI Companion, Measures, and Analysis Strategy

The GenAI support was implemented as a customized case companion combining instructor-defined behavioral instructions with an instructor-curated knowledge base containing the case study, solution code, and step-by-step solution material. Its purpose was to support model understanding, implementation, debugging, and solution extraction while encouraging students to reason through the task rather than simply retrieving answers. To maximize accessibility, the instructor implemented two versions of the companion: one using ChatGPT[8] and one using Google Gemini,[9] so that students could use whichever platform they had access to. Both companions were built from the same behavioral instructions and knowledge base. The full behavioral instructions are provided in the GitHub Repository (https://github.com/ceche1212/GenAI_for_OR_education).

Data were collected from four sources. First, the presurvey measured students' prior background, including previous decision-modeling performance, OR modeling proficiency, programming proficiency, LLM proficiency, verification habits, and perceived value of OR. Second, the postsurvey measured perceived learning gains, confidence, GPT usefulness, instructor and integrated support, task manageability, perceived difficulty, GPT-related risks, satisfaction, and open-ended feedback. Several items were deliberately paired across the two surveys to estimate changes in confidence and self-perceived capability. Third, students self-reported

Figure 3. Timeline of the Classroom Intervention



Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

performance indicators, including whether they solved the case, their best objective value, and the number of attempts. Fourth, open-ended questions captured what helped learning most, what was most difficult or least helpful, and how the companion was useful or not. The full questionnaires are provided in Online Appendix C.

The analysis strategy was designed to test a central hypothesis derived from the framework: that the value of GenAI support in OR education lies not in making tasks easy, but in making them manageable enough for students to remain productively engaged. This leads to a proposed mechanism in which perceived GPT usefulness and integrated instructor-GPT support are associated with task manageability, and manageability in turn drives perceived learning, confidence gain, and satisfaction. Critically, the hypothesis predicts that difficulty itself should not be the key driver of these outcomes: A task can be hard and still produce strong learning and satisfaction, provided the support structure makes it feel achievable.

The analysis proceeds in four steps. First, we examine pre-post confidence changes across the five paired items to assess whether the intervention was associated with meaningful gains in the areas most relevant to OR implementation. Second, we estimate regression models to identify the predictors of perceived learning gain and satisfaction, testing in particular whether task manageability explains outcomes beyond GPT usefulness and instructor support alone. Third, we conduct exploratory mediation analyses to test whether manageability mediates the relationship between support variables and learning outcomes, and whether this indirect pathway is stronger than any direct effect. Fourth, we estimate a compact observed-variable path model that jointly tests the full proposed mechanism, from support inputs through manageability to learning, confidence, and satisfaction while controlling for baseline proficiency and difficulty.

Because both class groups received the same pedagogical intervention, the analysis cannot establish causal effects. Designating one group as a control would have required withholding the GenAI-supported experience from students in the same course, which we considered educationally unjustifiable. The quantitative results are therefore interpreted as evidence on associations and mechanisms within the intervention rather than as causal estimates. To complement and contextualize these results, open-ended responses were analyzed using a deductive-inductive thematic coding procedure, with initial codes derived from the framework and refined iteratively during review. The qualitative analysis identifies students' perceived bottlenecks, benefits, and limitations of the GenAI-supported workflow, and is used to triangulate the quantitative findings rather than to generalize beyond this sample.

5. Classroom Evaluation Results

This section reports the results of the classroom evaluation. We first describe the sample, construct reliability, and activity outcomes, then examine pre-post confidence gains, predictors of perceived learning and satisfaction, mediation patterns, and the compact path model. We close by using open-ended responses to contextualize the quantitative findings and identify the main student-perceived benefits and difficulties of the GenAI-supported workflow.

5.1. Sample, Reliability, and Activity Outcomes

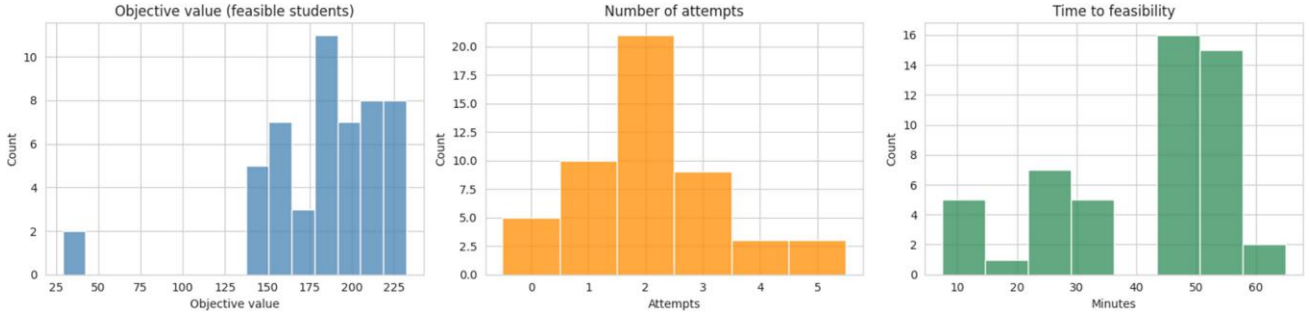
The final matched sample included 66 of the 73 enrolled students. Students entered the activity with moderate prior OR proficiency but limited programming experience and virtually no prior exposure to Pyomo. Their average decision-modeling grade from the previous semester was 12.62 out of 20, indicating a heterogeneous but generally adequate baseline in OR concepts. Full descriptive statistics are reported in Online Appendix D.

Survey variables were constructed as composite indices, each computed as the mean of its constituent items when at least 70% of items were available for a given student, with reverse-coded items recoded before aggregation. Presurvey indices captured OR proficiency, programming proficiency, LLM proficiency, perceived OR value, and verification tendency, where higher values on the verification index indicate more careful checking of GenAI outputs. Postsurvey indices captured perceived learning gain, difficulty and effort, GPT usefulness, GPT risk and overreliance, and instructor support. Task manageability and integrated support were single-item measures. Baseline confidence was the mean of the five paired presurvey confidence items, and confidence gain was the mean of the five corresponding postpre differences. Full item wording is in Online Appendix C and reliability statistics in Online Appendix D.

The activity produced a strong completion rate. Overall, 52 students self-reported solving the case, and 52 students (77.3% of the matched sample) reported a positive feasible solution after removing one clear objective-value outlier. As shown in Figure 4, the best observed objective value was 232.20, and the average standardized solution quality was 0.605. Among feasible students, the average time to feasibility was approximately 41 minutes with a median of two attempts. The task was therefore challenging but achievable within the classroom session.

Students engaged actively with the full support structure, reporting moderate use of the podcast, the instructor's introductory example, and the case text, and higher use of the GPT companion. This pattern matters because the intervention was not designed

Figure 4. Distribution of Objective Value, Number of Attempts, and Time to Feasibility



around unguided chatbot use: its value was hypothesized to come from the integrated combination of preparatory material, instructor scaffolding, case text, validation interface, and AI companion, not from any single element in isolation.

Construct reliability is reported in Online Appendix D. Reliability was acceptable for most constructs, particularly given that several scales were newly developed for this intervention. It was strongest for OR proficiency, programming proficiency, OR value, perceived learning gain, and GPT usefulness. Constructs with lower reliability, including verification tendency, difficulty and effort, GPT risk, and instructor support, are interpreted more cautiously in the analyses below.

A first indication of the proposed mechanism is visible in the correlation matrix reported in Figure 12 and Table 12 in Online Appendix D. Task manageability is positively correlated with GPT usefulness, integrated support, learning gain, confidence gain, feasibility, and satisfaction. Learning gain is positively correlated with confidence gain and satisfaction. Crucially, difficulty and effort are not meaningfully associated with satisfaction, which is an early signal that the intervention worked through manageability rather than through task ease. The following subsections test this mechanism directly.

5.2. Pre-Post Confidence Gains

The paired pre-post items show clear confidence gains across all five dimensions assessed. The largest increases occurred precisely where students entered with the lowest baseline confidence: programming comfort, implementation of mathematical models in Python, and debugging. Smaller but still significant gains were observed for explaining OR model components and for using GenAI as a learning support tool. Figure 5 summarizes the item-level distributions.

Table 4 reports the paired comparisons. All item-level changes were statistically significant under both paired t-tests and Wilcoxon signed-rank tests. The largest standardized gains were for comfort in Python ($d_z = 1.23$), implementation of mathematical models in Python ($d_z = 1.13$), and debugging Python errors ($d_z = 0.88$), all large effects by conventional standards.

At the composite level, the mean confidence gain across the five items was 0.945 points on the five-point scale ($SD = 0.736$), significantly greater than zero, $t(65) = 10.44$, $p < 0.001$, confirmed by the Wilcoxon signed-rank test ($W = 91.0$, $p < 0.001$), with a large standardized effect ($d_z = 1.29$). These gains are concentrated in the computational stages of OR modeling, precisely the transitions from mathematical formulation to executable code that represent the most persistent bottleneck for business students encountering Python-based optimization for the first time.

5.3. Perceived Learning, Manageability, and Support

This section tests the central mechanism of the framework: that GenAI and instructor support improve learning not simply by being available, but by making a complex OR task manageable. The hypothesized pathway is support structure → perceived manageability → perceived learning gain. We first estimate a regression model predicting perceived learning gain and then use mediation analyses to test whether manageability explains the associations between support variables and learning.

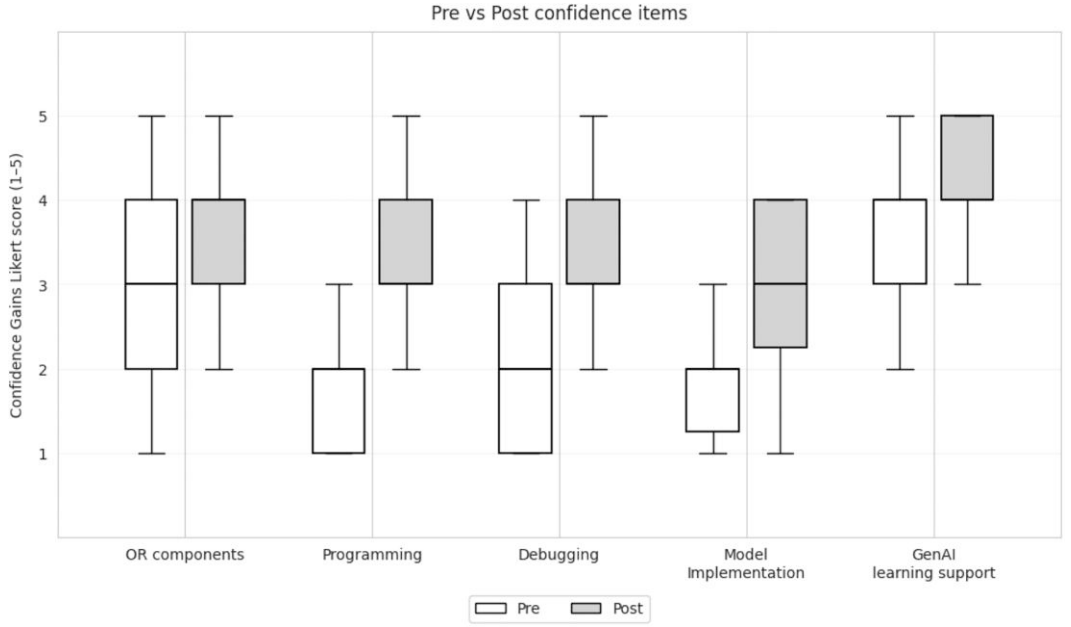
The regression results in Table 5 are unambiguous: Task manageability is the strongest and only significant predictor of perceived learning gain ($\beta = 0.366$, $p < 0.001$). Neither GPT usefulness nor instructor support reached significance once manageability and baseline controls were included. Programming proficiency was positive but only marginally significant ($\beta = 0.112$, $p = 0.072$). The key implication is that perceived learning was not driven by students simply rating the GPT as useful or the instructor as helpful. What mattered was whether the combined support structure made the task feel achievable.

Two mediation models test this mechanism directly. The first asks whether GPT usefulness reaches learning through manageability. GPT usefulness was positively associated with manageability ($\beta = 0.453$, $p < 0.001$),

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

Figure 5. Pre- and Postintervention Confidence Items



Note. Boxplots show the distribution of Likert-scale responses before and after the activity for the five paired items.

and manageability was strongly associated with learning gain ($\beta = 0.643$, $p < 0.001$). Once manageability was included, the direct path from GPT usefulness to learning gain became nonsignificant ($\beta = 0.081$, $p = 0.500$), whereas the indirect effect through manageability was significant ($\beta = 0.273$, $p = 0.001$). GPT usefulness contributed to learning primarily by helping students experience the task as manageable, not by directly producing learning on its own.

The second mediation asks whether the broader integrated support structure operates through the same channel. Integrated support was positively associated with manageability ($\beta = 0.394$, $p = 0.001$), and the indirect effect on learning through manageability was again significant ($\beta = 0.233$, $p = 0.006$). By contrast, instructor support alone showed only a marginal mediation pattern. This distinction matters: The learning mechanism was not isolated instructor help or isolated GPT help, but their integration with the case material, the prepared notebook, and the validation interface into a coherent learning environment.

Taken together, these results support the main design claim of the intervention. The GenAI-supported workflow improved perceived learning not by making the task easy or by replacing student effort, but by making a genuinely difficult OR implementation task manageable enough for students to remain productively engaged. Full mediation outputs are reported in Online Appendix E.

5.4. Confidence, Satisfaction, and the Role of Difficulty

The results for confidence gain and satisfaction reinforce the manageability mechanism and add an important clarification: Neither outcome was driven by task ease.

For confidence gain, the dominant predictor was perceived learning gain ($\beta = 0.375$, $p < 0.001$), not task performance. Baseline confidence was strongly and negatively associated with confidence gain ($\beta = -0.484$, $p < 0.001$), meaning students who entered with the lowest confidence reported the largest increases. GPT usefulness showed a small positive association in the baseline model ($\beta = 0.114$, $p = 0.027$) but became nonsignificant once manageability and learning were included, indicating that any direct effect of the

Table 4. Paired Pre-Post Confidence Comparisons

Paired item	N	Pre M	Post M	Δ	t	d_z
Explain OR components	66	3.06	3.53	0.47	4.61	0.57
Comfort in Python	66	1.91	3.29	1.38	10.00	1.23
Debug Python errors	66	2.05	3.15	1.11	7.16	0.88
Implement math models in Python	66	1.94	3.02	1.08	9.20	1.13
Use GenAI as learning support	66	3.48	4.18	0.70	5.38	0.66

Note. All paired t-tests and Wilcoxon signed-rank tests were significant at $p < 0.001$.

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

Table 5. Regression Model Predicting Perceived Learning Gain

Predictor	β	Standard error	p
GPT usefulness	0.047	0.093	0.615
Instructor support	0.050	0.070	0.479
Task manageability	0.366	0.100	<0.001
Difficulty/effort	0.007	0.076	0.929
OR proficiency	−0.004	0.088	0.967
Programming proficiency	0.112	0.062	0.072
LLM proficiency	−0.030	0.092	0.745
Verification tendency	−0.147	0.095	0.121

Note. Coefficients are standardized and robust standard errors are reported.

companion on confidence was fully explained by its contribution to learning. Solution quality was not associated with confidence gain at any stage. The practical implication is direct: What built confidence was not solving the case well but perceiving that learning had occurred.

For satisfaction, perceived learning gain was again the dominant predictor ($\beta = 0.343$, $p < 0.001$), with task manageability also emerging as a significant independent predictor when included ($\beta = 0.167$, $p = 0.044$). Critically, difficulty was unrelated to satisfaction in every model estimated ($\beta = -0.015$, $p = 0.814$). Students were not more satisfied because the task was easy; they were satisfied because the support structure made it manageable and because they perceived themselves as having learned.

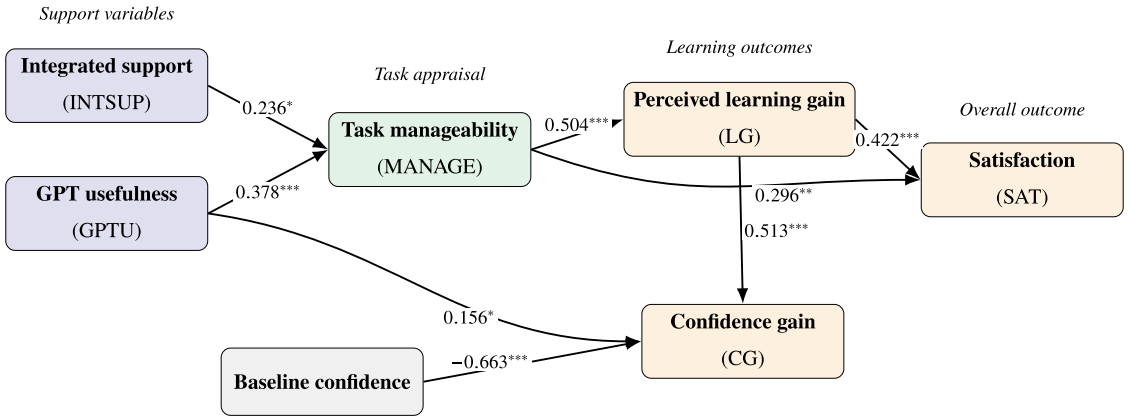
The mediation analysis tests whether the pathway manageability → learning gain → satisfaction explains how the support structure produced positive outcomes. Manageability was strongly associated with learning gain ($\beta = 0.644$, $p < 0.001$), learning gain was strongly associated with satisfaction ($\beta = 0.617$, $p < 0.001$), and the indirect effect of manageability on satisfaction through learning was significant ($\beta = 0.285$, $p < 0.001$). Manageability also retained a significant direct effect on satisfaction ($\beta = 0.314$, $p = 0.019$), indicating that it contributed to satisfaction both by increasing perceived learning and independently of it. Taken together, these results paint a consistent picture: the intervention succeeded not by removing difficulty, but by making a hard task feel worthwhile. Full regression and mediation outputs are reported in Online Appendix E.

5.5. Mediation and Compact Path Model

The regression and mediation results above each test one piece of the proposed mechanism in isolation. The compact path model tests whether the full chain holds simultaneously: that GPT usefulness and integrated support predict manageability, manageability predicts learning, and learning predicts confidence and satisfaction, whereas difficulty remains irrelevant throughout. The model was estimated on standardized variables and includes baseline confidence and proficiency controls. Figure 6 displays the significant structural paths and Table 6 reports the full set of estimates.

The model tells a coherent and consistent story. At the input stage, both GPT usefulness ($\beta = 0.378$, $p < 0.001$) and integrated support ($\beta = 0.236$, $p = 0.022$) are positively associated with task manageability, confirming that the support structure, taken as a whole, shaped how students experienced the task. Manageability in turn was strongly associated with perceived learning gain ($\beta = 0.504$, $p < 0.001$), whereas the direct path from GPT usefulness to learning gain was not significant once manageability was included ($\beta = 0.108$, $p = 0.358$). This is a critical result: The companion did not improve learning by giving students answers. It improved learning by making a difficult task feel tractable.

Figure 6. Compact Exploratory Path Model for the Main Support Mechanism



Notes. Standardized coefficients are shown on the arrows. To preserve readability, the figure displays only the focal constructs and significant structural paths. The full estimated model, including additional controls and covariances, is reported in the corresponding results table.

Significance levels: $^{*}p < 0.05$; $^{**}p < 0.01$; $^{***}p < 0.001$.

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

Table 6. Main Standardized Paths in the Compact Path Model

Path	β	p
Task manageability ← GPT usefulness	0.378**	<0.001
Task manageability ← Integrated support	0.236*	0.022
Learning gain ← Task manageability	0.504**	<0.001
Learning gain ← GPT usefulness	0.108	0.358
Learning gain ← Difficulty	0.030	0.773
Confidence gain ← Learning gain	0.513**	<0.001
Confidence gain ← Baseline confidence	−0.663**	<0.001
Confidence gain ← GPT usefulness	0.156*	0.028
Satisfaction ← Learning gain	0.422**	<0.001
Satisfaction ← Task manageability	0.296**	0.006
Satisfaction ← Confidence gain	0.091	0.382
Satisfaction ← Difficulty	−0.024	0.792

The downstream paths are equally clear. Learning gain was strongly associated with confidence gain ($\beta = 0.513$, $p < 0.001$), with the largest gains accruing to students who entered with the lowest baseline confidence ($\beta = -0.663$, $p < 0.001$). Learning gain was also strongly associated with satisfaction ($\beta = 0.422$, $p < 0.001$), and manageability retained an independent direct effect on satisfaction ($\beta = 0.296$, $p = 0.006$), indicating that the experience of working through a manageable task was itself rewarding, beyond any learning it produced. Throughout the entire model, difficulty was unrelated to both learning gain ($\beta = 0.030$, $p = 0.773$) and satisfaction ($\beta = -0.024$, $p = 0.792$). Students were not more satisfied or more confident because the task was easy. They were satisfied and confident because they learned.

The compact path model therefore does more than corroborate the earlier regression and mediation results. It demonstrates that the proposed mechanism holds as a coherent system: support inputs predict manageability, manageability drives learning, and learning drives confidence and satisfaction, with difficulty playing no role at any stage. For OR education, this is a meaningful finding. It suggests that the right question when designing GenAI-supported learning is not how to make OR tasks easier, but how to make them manageable enough that students can engage, persist, and ultimately learn. Full path estimates and model fit statistics are reported in Online Appendix E.

5.6. Qualitative Findings

The open-ended responses provide direct evidence on how students experienced the GenAI-supported workflow. Of the 66 matched students, 62 responded to the question on what helped learning most, 62 to what was most difficult or least helpful, and 59 to the usefulness or limitations of the GPT companion. Responses were coded using a deductive-inductive thematic procedure, with initial codes derived from the framework and refined during review. The full codebook, coded matrix, theme frequencies, and quote bank are reported in Section E.3 in Online Appendix E. Three findings stand out, each directly corroborating a quantitative result.

5.6.1. Implementation Was the Dominant Bottleneck. The most frequent theme across all responses was coding, Python, Pyomo, or Colab difficulty, appearing in 80 coded segments and dominating responses about what was most difficult (40 coded responses). Students described challenges such as "Understand how to code," "Find errors in the code," and "Clearly understand how to write on Google Colab." This confirms that the intervention targeted a genuine and severe bottleneck: not understanding the model conceptually but translating it into a working computational implementation.

5.6.2. GPT Companion Functioned as a Step-by-Step Guide, Not an Answer Provider. The theme *GPT as step-by-step tutor and dialogue* appeared in 52 coded segments, including 23 responses to what helped learning most and 19 to how the companion was useful. Students described it as something that "could explain step by step" and that allowed "easy Q & A" and clues. This is precisely the manageability mechanism identified quantitatively: the companion helped students move through intermediate modeling, coding, and debugging steps rather than delivering final solutions.

5.6.3. Students Valued the Integrated Support Structure, Not the AI Tool in Isolation. Case and context understanding appeared in 45 coded segments, and instructor scaffolding and debrief in 15. Students cited "the professor's classes," "the teacher explanations," and the value of working on "a real problem." These responses reinforce the path model finding that integrated support, combining instructor preparation, case context, validation interface, and GPT assistance, predicted manageability more strongly than any single element alone.

The responses also surface important limitations that must not be overlooked. GPT limitations and friction appeared in 23 coded segments, with students reporting limited chat availability, frustrating interactions, and imperfect guidance. A smaller but critical set of responses highlighted the difficulty of "understanding if the code given by AI was actually correct or not," pointing directly to the verification challenge that the framework identifies as an instructor responsibility. These limitations do not undermine the quantitative findings, but they reinforce that GenAI support in OR education requires deliberate guardrails, instructor oversight, and explicit training in critical verification. The qualitative evidence therefore does not simply echo the numbers: It explains the mechanism from the student's perspective, and it names the conditions under which that mechanism can break down.

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

6. Discussion and Conclusions

This article set out to answer a practical question: can generative AI be integrated into OR education in a way that genuinely supports learning, rather than simply making tasks easier or replacing student reasoning? The evidence suggests that it can, provided it is designed as targeted instructional support rather than deployed as a generic answer engine.

The contribution is threefold. We developed a task-centered framework mapping GenAI support functions to the main cognitive bottlenecks of OR learning. We translated that framework into a set of use cases and a staged adoption roadmap. Also, we evaluated one classroom implementation through pre-post survey data, self-reported performance indicators, mediation and path analyses, and open-ended student feedback, providing mixed-methods evidence on how and why the intervention worked.

6.1. What the Evidence Shows

The central finding is that the intervention worked through manageability. Students faced a genuinely demanding task: understanding a realistic sprint-planning case, translating it into a quadratic mixed-integer programming model, implementing it in Python with Pyomo and HiGHS, extracting the solution, and validating feasibility. The support structure made this complex workflow achievable for 77.3% of students within a two-hour session. Task manageability was consistently associated with perceived learning gain, confidence gain, and satisfaction across every model estimated. GPT usefulness and integrated support predicted manageability; manageability predicted learning; learning predicted confidence and satisfaction. Difficulty was irrelevant throughout.

The pre-post confidence gains reinforce this picture. The largest gains occurred in programming comfort ($d_z = 1.23$), implementation of mathematical models in Python ($d_z = 1.13$), and debugging ($d_z = 0.88$), precisely the computational transitions where business students most commonly stall when OR moves from formulation to executable code. These are not cosmetic improvements in attitude. They represent meaningful gains in the specific capabilities that OR practice demands.

The role of the GenAI companion deserves precise characterization. GPT usefulness did not directly predict learning once manageability was included in the model. This is not a negative finding: it means the companion worked as intended. Its value was not that it provided answers, but that it helped students progress through intermediate steps, understanding the model, translating notation into code, diagnosing errors, and extracting the solution, in a way that made a difficult task feel tractable. The qualitative evidence confirms this: Students described the companion as a step-by-step guide rather than a solution provider, and the most frequent theme across all open-ended responses was implementation difficulty and not AI over-reliance.

6.2. Implications for or Instructors

The practical implication is to start from the bottleneck and not the technology. Audio and narrative supports help when the bottleneck is contextualization. Flashcards and retrieval practice help when the bottleneck is retention of modeling patterns. Customized case companions help when the bottleneck is formulation, implementation, or debugging. Retrieval-augmented systems help when the bottleneck is access to specialized documentation or domain knowledge. The design question is not "Should students use GenAI?" but rather "At which point in the OR workflow can GenAI help students remain productively engaged without bypassing the reasoning task?"

Three broader design principles follow from the evidence. First, GenAI should preserve cognitive integrity by providing hints, explanations, and partial guidance rather than replacing student reasoning (Bastani et al. 2025). Second, the instructor is not optional. GenAI can reduce repetitive support demands, but the instructor remains essential for structuring the task, validating reasoning, and leading reflection. OR models function as microworlds for testing assumptions and refining mental models (Morecroft 1992): that reflective function cannot be delegated to an AI. Third, critical evaluation of AI outputs must be taught explicitly (Jackson 2001). In quantitative domains, plausible but incorrect formulations or code fragments can mislead students precisely because they look reasonable. Verification is a skill, not a default behavior.

The findings also point to a concrete opportunity: GenAI-supported instruction may allow OR educators to introduce more realistic problem variants without overwhelming students. Basic models such as the knapsack problem are pedagogically necessary, but real decision problems are rarely so clean. Multiperiod, quadratic, precedence-constrained, or multiresource formulations are closer to professional practice but have traditionally been out of reach in a single classroom session. The results of this study suggest that a well-designed AI companion can bridge that gap, making richer formulations manageable without sacrificing rigor. Finally, equitable adoption requires attention to access. Commercial frontier models perform well, but open-source alternatives, shared teaching resources, and curated knowledge bases are essential for making GenAI-supported OR education broadly available (Reisman 2004).

6.3. Limitations and Future Research

The study has important limitations that future work must address. The sample comprised 66 students in

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

two class groups within a single institution, and the design did not include a randomized control group. The analysis therefore establishes associations and mechanisms, not causal effects. Several constructs are self-reported, some survey scales were newly developed and showed modest reliability, and the study captures short-term outcomes after a single case activity rather than long-term retention or transfer.

Future research should replicate this design across different institutions, instructors, cohorts, and OR topics. Stronger evaluation designs could incorporate staggered rollouts, crossover designs, objective performance tests, notebook and validation-system logs, and delayed retention assessments. Comparative studies examining generic chatbots against customized tutors, retrieval-augmented assistants, and tightly constrained case companions would also help clarify which design choices matter most. A particularly important open question is transfer: whether students who receive AI-supported help on one modeling task become better able to formulate, implement, and validate a different problem independently. Future work should also explore how anonymized student questions, common implementation errors, and validated solution explanations can be used responsibly to build more grounded, OR-specific retrieval-augmented systems, subject to appropriate governance, consent, and instructor oversight. A fuller discussion of ethical and governance considerations is provided in Online Appendix F.

6.4. Conclusion

OR education faces a persistent tension: The field demands technical rigor, abstract reasoning, and computational skill, yet these are precisely the qualities that make it difficult to teach at scale and difficult for students to develop without sustained support. This study suggests that generative AI, when designed as instructional support rather than as an answer engine, can meaningfully reduce that tension. Its value is not in making OR easy. It is in making difficult modeling and implementation tasks manageable enough for students to engage, persist, debug, and ultimately understand. For a field whose educational reach has long lagged behind its professional impact, that is a meaningful step forward.

Acknowledgments

We thank Verena Ehrler for contributions and feedback to this research.

Endnotes

[1] The repository includes prompts, examples, and implementation material for the GenAI-supported OR learning activities discussed in this article.

[2] For an example of an agentic system interacting with a solver through autonomous tool calls, see https://www.youtube.com/watch?v=tYg0LHnri2w.

[3] A user story is a short, plain-language description of a software feature written from the perspective of the end user, commonly used in agile development to define the scope of work for a development team.

[4] A sprint is a fixed-length iteration in agile project management, typically lasting one to four weeks, during which a development team completes a defined set of work items.

[5] Pyomo is an open-source algebraic modeling language for mathematical programming implemented in Python. It allows users to define optimization models using familiar Python syntax and connect them to a wide range of solvers.

[6] HiGHS is an open-source, high-performance solver for linear programming, mixed-integer programming, and quadratic programming problems. It is freely available and can be called directly from Python, making it well suited for classroom use.

[7] Google Colab is a free, cloud-based Jupyter notebook environment provided by Google. It requires no local installation and allows users to write and execute Python code directly in a web browser, making it particularly accessible for classroom settings where students have heterogeneous computing setups.

[8] The ChatGPT companion is publicly accessible at https://chatgpt.com/g/g-69dd14d61880819191f6f8ee938f49ac-agile-planning-at-synapseflow.

[9] The Gemini companion is publicly accessible at https://gemini.google.com/gem/1KeIxoZJ3QnrgLmhOmDu1q9Ckh_cI1Cp7?usp=sharing.

References

Artigues C, Demassey S, Néron E (2008) *Resource Constrained Project Scheduling: Models, Algorithms, Extensions and Applications* (ISTE Ltd., London, UK; John Wiley & Sons, Hoboken, NJ).

Bastani H, Bastani O, Sungu A, Ge H, Kabakcı Ö, Mariman R (2025) Generative AI without guardrails can harm learning: Evidence from high school mathematics. *Proc. Natl. Acad. Sci. USA* 122(26):e2422633122.

Beliën J, De Boeck L, Goossens D, Quttineh NH, Vermeir J (2025) Game—The game of thrones quest for optimality: A role playing game for teaching the art of linear programming. *INFORMS Trans. Ed.* 25(3):250–256.

Bennett KP, Parrado-Hernández E (2006) The interplay of optimization and machine learning research. *J. Machine Learn. Res.* 7(46): 1265–1281.

Bertsekas DP (2025) Neuro-dynamic programming. Pardalos PM, Prokopyev OA, eds. *Encyclopedia of Optimization* (Springer, New York), 1–6.

Bien J, Mukherjee G (2025) Generative AI for data science 101: Coding without learning to code. *J. Statist. Data Sci. Ed.* 33(2):129–142.

Boroughani T, Behshad N, Xodabande I (2023) Mobile-assisted academic vocabulary learning with digital flashcards: Exploring the impacts on university students' self-regulatory capacity. *Frontiers Psych.* 14:1112429.

Boschetti MA, Golfarelli M, Rizzi S, Turricchia E (2014) A lagrangian heuristic for sprint planning in agile software development. *Comput. Oper. Res.* 43:116–128.

Boutilier JJ, Chan TC (2023) Introducing and integrating machine learning in an operations research curriculum: An application-driven course. *INFORMS Trans. Ed.* 23(2):64–83.

Boyd SP, Vandenberghe L (2004) *Convex Optimization* (Cambridge University Press, Cambridge, UK).

Brandon-Jones A, Piercy N, Slack N (2012) Bringing teaching to life: Exploring innovative approaches to operations management education. *Internat. J. Oper. Production Management* 32(12):1369–1374.

Bray RL (2025) A tutorial on teaching data analytics with generative ai. *INFORMS J. Appl. Anal.* 55(4):319–343.

Bureau of Labor Statistics (2025) Occupational outlook handbook: Operations research analysts. Accessed July 12, 2025, https://www.bls.gov/ooh/math/operations-research-analysts.htm.

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.

Cribben I, Zeinali Y (2023) The benefits and limitations of chatgpt in business education and research: A focus on management science, operations management and data analytics. Preprint, submitted April 3, https://doi.org/10.2139/ssrn.4404276.

De Simone ME, Tiberti FH, Rodriguez MRB, Manolio FA, Mosuro W, Dikoru EJ (2025) From chalkboards to chatbots: Evaluating the impact of generative ai on learning outcomes in nigeria. Policy Research Working Paper No. 11125, The World Bank, Washington, DC.

Dean MD (2020) Using the learning assistant model in an undergraduate business analytics course. *INFORMS Trans. Ed.* 20(3):125–133.

Demeulemeester EL, Herroelen WS (2002) *Project Scheduling: A Research Handbook* (Springer, New York).

DePuy GW, Taylor GD (2007) Using board puzzles to teach operations research. *INFORMS Trans. Ed.* 7(2):160–171.

Deslauriers L, Schelew E, Wieman C (2011) Improved learning in a large-enrollment physics class. *Science (1979)* 332(6031):862–864.

Dias J (2017) Teaching operations research to undergraduate management students: The role of gamification. *Internat. J. Management Ed.* 15(1):98–111.

Drake MJ (2019) Teaching OR/MS with cases: A review and new suggestions. *INFORMS Trans. Ed.* 19(2):57–66.

Ebbinghaus H (1913) *Memory: A Contribution to Experimental Psychology* (Teachers College, Columbia University, New York).

Ellis AR, Slade E (2023) A new era of learning: Considerations for chatGPT as a tool to enhance statistics and data science education. *J. Statist. Data Sci. Ed.* 31(2):128–133.

Fang K, Tang C, Wang J (2025) Evaluating simulated teaching audio for teacher trainees using RAG and local LLMs. *Sci. Rep.* 15(1):3633.

Frances DM, Terekhov D (2019) A case-based undergraduate operations research course. *INFORMS Trans. Ed.* 19(2):67–80.

Goffin K (1998) Operations management teaching on European MBA programmes. *Internat. J. Oper. Production Management* 18(5):424–451.

Grinde RB, Kammermeyer JA (2003) Experiences using thematic assignments in an undergraduate management science course. *INFORMS Trans. Ed.* 4(1):23–36.

Gunderson JL, Cumming TM (2023) Podcasting in higher education as a component of universal design for learning: A systematic review of the literature. *Innovations Ed. Teaching Internat.* 60(4):591–601.

Hammedi W, Leclercq T, Poncin I, Alkire L (2021) Uncovering the dark side of gamification at work: Impacts on engagement and well-being. *J. Bus. Res.* 122:256–269.

Horowitch R (2024) The elite college students who can't read books. *The Atlantic* (November), https://www.theatlantic.com/magazine/archive/2024/11/the-elite-college-students-who-cant-read-books/679945/.

Jackson MC (2001) Critical systems thinking and practice. *Eur. J. Oper. Res.* 128(2):233–244.

Joseph-Richard P, Uhomoibhi J (2024) Which data sets are preferred by university students in learning analytics dashboards? A situated learning theory perspective. *INFORMS Trans. Ed.* 24(3):220–237.

Kosmyna N, Hauptmann E, Yuan YT, Situ J, Liao XH, Beresnitzky AV, Braunstein I, et al. (2025) Your brain on chatGPT: Accumulation of cognitive debt when using an AI assistant for essay writing task. Preprint, submitted June 10, https://arxiv.org/abs/2506.08872.

Kwon HY, Özpolat K (2021) The dark side of narrow gamification: Negative impact of assessment gamification on student perceptions and content knowledge. *INFORMS Trans. Ed.* 21(2): 67–81.

Latimier A, Peyre H, Ramus F (2021) A meta-analytic review of the benefit of spacing out retrieval practice episodes on retention. *Ed. Psych. Rev.* 33(3):959–987.

Le TV, Albert LA, Vidal T (2025) Making operations research more accessible: Insights from the rise of machine learning. *INFORMS J. Data Sci.* 5(1):1–13.

Liebman JS (1998) Teaching operations research: Lessons from cognitive psychology. *Interfaces (Providence)* 28(2):104–110.

Mollick E, Mollick L (2022) New modes of learning enabled by ai chatbots: Three methods and assignments. Preprint, submitted December 23, https://doi.org/10.2139/ssrn.4300783.

Morecroft JD (1992) Executive knowledge, models and learning. *Eur. J. Oper. Res.* 59(1):9–27.

National Center for Education Statistics (2023) The Nation's report card: Long-term trend assessment highlights: Reading and mathematics at age 13. Technical report, U.S. Department of Education, Institute of Education Sciences, Washington, DC.

National Endowment for the Arts (2022) The 2020 Arts Basic Survey. Technical report, National Endowment for the Arts, Washington, DC.

Nestler S, Levis J, Klimack B (2012) Certified analytics professional: Informs prepares to launch first-of-its-kind program. *OR/MS Today* 39(5):16–21.

Pardo EG, Gil-Borrás S, Alonso-Ayuso A, Duarte A (2024) Order batching problems: Taxonomy and literature review. *Eur. J. Oper. Res.* 313(1):1–24.

Pérez Armas LF (2024) Resource planning at l'audobarquoise of saint-omer: A graph theory and mathematical programming case study with python. Technical report No. 624-0030-4, The Case Centre, Cranfield, Bedfordshire, UK.

Reisman A (2004) How can or/ms educators benefit from creating and using taxonomies? *INFORMS Trans. Ed.* 4(3):55–65.

Sánchez JMG (2022) Building and solving mathematical programming models: 50 Practical Examples. *International Series in Operations Research & Management Science*, vol. 329 (Springer, Cham, Switzerland).

Scheubrein RJ, Kulturel-Konak S (2006) An electronic teaching assistant for basic operations management methods. *INFORMS Trans. Ed.* 7(1):99–105.

Schmidhuber J (2015) Deep learning in neural networks: An overview. *Neural Networks* 61:85–117.

Schwindt C, Zimmermann J (2015a) *Handbook on Project Management and Scheduling*, vol. 1 (Springer International Publishing, Cham, Switzerland).

Schwindt C, Zimmermann J, eds. (2015b) *Handbook on Project Management and Scheduling, Volume 2 of International Handbooks on Information Systems* (Springer).

Shojaee P, Mirzadeh I, Alizadeh K, Horton M, Bengio S, Farajtabar M (2025) The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity. Preprint, submitted June 10, https://arxiv.org/abs/2506.06941.

Simon HA (1987) Two heads are better than one: The collaboration between AI and OR. *Interfaces (Providence)* 17(4):8–15.

Singh A, Alexander PA (2022) Audiobooks, print, and comprehension: What we know and what we need to know. *Ed. Psych. Rev.* 34(2):677–715.

Smith CD, Scarf D (2017) Spacing repetitions over long timescales: A review and a reconsolidation explanation. *Frontiers Psych.* 8:962.

Smith MK, Wood WB, Adams WK, Wieman C, Knight JK, Guild N, Su TT (2009) Why peer discussion improves student performance on in-class concept questions. *Science (1979)* 323(5910): 122–124.

Smolen P, Zhang Y, Byrne JH (2016) The right time to learn: Mechanisms and optimization of spaced learning. *Nature Rev. Neurosci.* 17(2):77–88.

Sutton RS, Precup D, Singh S (1999) Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence* 112(1–2):181–211.

Szilard L (1964) On memory and recall. *Proc. Natl. Acad. Sci. USA* 51(6):1092–1099.

Van den Broeke M, Mlinar T (2025) The competitive car production game: Teaching theory of constraints and linear programming. *INFORMS Trans. Ed.* 26(3):233–246.

Van Hentenryck P, Bent R (2006) *Online Stochastic Combinatorial Optimization* (The MIT Press, Cambridge, MA).

Vanhoucke M. (2012) *Project Management with Dynamic Scheduling* (Springer, Berlin, Heidelberg, Germany).

Williams HP (2013) *Model Building in Mathematical Programming* (John Wiley & Sons, Hoboken, NJ).

Xu X (2025) Ai optimization algorithms enhance higher education management and personalized teaching through empirical analysis. *Sci. Rep.* 15(1):10157.

Downloaded from informs.org by [216.73.216.221] on 10 November 2026, at 15:47 . For personal use only, all rights reserved.