Proceedings of the 2025 Winter Simulation Conference
E. Azar, A. Djanatliev, A. Harper, C. Kogler, V. Ramamohan, A. Anagnostou, and S. J. E. Taylor, eds.

PERFORMANCE OF LLMS ON STOCHASTIC MODELING OPERATIONS RESEARCH PROBLEMS: FROM THEORY TO PRACTICE

Akshit Kumar[1], Tianyi Peng[1], Yuhang Wu[1], and Assaf Zeevi[1]

[1]Columbia Business School, Columbia University, New York, NY, US

ABSTRACT

Large language models (LLMs) have exhibited capabilities comparable to those of human experts in various fields. However, their *modeling abilities*—the process of converting real-life problems (or their verbal descriptions) into sensible mathematical models—have been underexplored. In this work, we take the first step to evaluate LLMs' abilities to solve stochastic modeling problems, a model class at the core of Operations Research (OR) and decision-making more broadly. We manually procure a representative set of graduate-level homework and doctoral qualification-exam problems and test LLMs' abilities to solve them. We further leverage `SimOpt`, an open-source library of simulation-optimization problems and solvers, to investigate LLMs' abilities to make real-world decisions. Our results show that, though a nontrivial amount of work is still needed to reliably automate the stochastic modeling pipeline in reality, state-of-the-art LLMs demonstrate proficiency on par with human experts in both classroom and practical settings.

1 INTRODUCTION

Large language models (LLMs) have shown great potential in solving many tasks, such as language modeling, translation, and question answering. More recently, the advent of reasoning models, such as GPT-o1 (OpenAI 2025) and DeepSeek-R1 (Guo et al. 2025), has further pushed LLMs' ability to solve math problems. In general, there are two popular types of benchmarks relevant to this task. The first one usually considers problems that have unambiguous numerical answers, much like high-school math problems or math competition problems; example datasets constructed for these purposes include MATH (Hendrycks et al. 2021) and GSM8K (Cobbe et al. 2021). The other type of benchmarks target formal theorem proving and often involve the use of functional programming languages such as Lean (Yang et al. 2023).

However, these types of benchmarks are not suitable for evaluating LLMs' ability to abstract more complex problem descriptions into a formal mathematical framework, a procedure often referred to as *modeling*. Modeling problems are ubiquitous in practice and are at the core of the field of Operations Research (OR). As a field, OR differentiates itself by its focus on the science and art of modeling, through which it aims to distill sensible mathematical models that can be analyzed for insights and structure of solutions. This process is highly nontrivial and usually requires a deep understanding of both the technical skills and the problem domain. An ideal model should be able to capture the essential features of the problem, while being simple enough to be solved analytically or numerically. Clearly, neither math-exam-style problems nor pure theorem-proving problems can adequately test LLMs' modeling abilities.

Meanwhile, one may argue that this modeling ability is crucial for an intelligent agent. By abstracting tangible phenomena into structured mathematical representations, an intelligent agent not only simplifies the inherent uncertainty of everyday problems but also enables systematic reasoning, simulation, and prediction. Recently, the OR community has begun to explore the potential of LLMs in solving modeling problems with an emphasis on optimization (Ramamonjison et al. 2022; AhmadiTeshnizi et al. 2024; Huang et al. 2025). As a central pillar of OR, optimization problems have a wide range of real-world applications, and the ability to properly formulate and solve them is a key skill for OR practitioners.

In contrast, the evaluation of LLMs' ability to handle *stochastic modeling* problems, another pillar of OR, has been mostly absent in the literature. Stochastic modeling problems cast the decision-making process in a framework of uncertainty, which is inherent to many real-world problems. Compared to optimization in a deterministic scenario, decision-making with uncertainty is intrinsically harder because a "good" decision has to work for a range of possible outcomes that are often unpredictable and may only be tractable en masse through an often unknown probability distribution. For example, consider stochastic processing networks (Dai and Harrison 2020), a well-known class of challenging stochastic modeling problems. Though some networks have seemingly simple structures, there is often no known analytical solution, and one must resort to numerical solutions, where a range of other issues (e.g., the choice of a solver) arise. In general, the complexity of a stochastic modeling problem scales up quickly as multiple sources of randomness become entangled, and a real-world problem can easily become overly challenging. If some specialized intelligent agent can be developed to possess human-expert-level capabilities, an incredible amount of potential can be unlocked just by the power of automation itself.

Therefore, as a first step, we believe there is an urgent need to assess LLMs' ability to solve these stochastic modeling problems, and hence add to our understanding of the LLMs' ability to perform modeling abstractions. Our contributions include:

- We construct a dataset of graduate-level stochastic modeling problems to test LLMs' abilities to solve them. We find strong performance overall with more open-ended modeling problems presenting a greater challenge.
- We procure a set of qualification-exam problems and manually grade LLMs' answers to them. Our results show that they demonstrate comparable performance to human PhD candidates in the field.
- Using the `SimOpt` library, we show that top-performing LLMs can match the best in-house solvers on a range of simulation-optimization problems, underscoring their potential. However, our results also indicate that off-the-shelf use of LLMs is not yet reliable for automating the full stochastic modeling pipeline, suggesting the need for further adaptation and integration.

In summary, our findings demonstrate that LLMs hold significant promise across stochastic modeling tasks—from analytical problem solving to simulation optimization—but realizing reliable end-to-end automation will require targeted refinement beyond off-the-shelf use. We hope this work encourages further research on automating stochastic modeling and contributes to building intelligent agents capable of autonomously understanding, modeling, and solving complex real-world problems.

The rest of the paper is organized as follows. In Section 2, we review the related literature. In Section 3, we evaluate LLMs' performance on the homework problems dataset. In Section 4, we assess LLMs' performance on the qualifying exam problems. In Section 5, we discuss the simulation-optimization problems and compare LLMs' solutions with standard solvers. Finally, in Section 6, we conclude the paper and discuss future directions.

2 LITERATURE REVIEW

LLMs have long been tested for their ability to solve math problems. GPT-3 was one of the first large-scale generative models to demonstrate strong zero-shot and few-shot capabilities across many tasks, including simple mathematical problems (Brown et al. 2020). Soon after, Hendrycks et al. (2021) introduced the MATH dataset, which consists of over 12,500 competition-level math problems covering algebra, geometry, number theory, and more, emulating the difficulty of middle and high-school math competitions. Concurrently, Cobbe et al. (2021) released the GSM8K dataset that contains over 8,000 short-answer math word problems (grade-school level) focusing on arithmetic and multi-step reasoning. They also proposed a "verifier" model trained to check correctness of solutions. MATH and GSM8K datasets became widely used benchmarks to test reasoning and multi-step solution correctness. Many later papers used them as primary testbed, including the renowned chain-of-thought paper (Wei et al. 2022) and subsequent papers

on LLM reasoning. It is evident by now that these standard datasets are too rudimentary to keep up with the problem-solving capabilities of newer models. Therefore, there is a need for new benchmarks with deeper symbolic reasoning and proof-like questions.

One direction that is more specialized and more challenging is applying LLMs to the stricter setting of formal logic and interactive theorem proving. Unlike natural language math solutions, where partial correctness is sometimes acceptable, formal theorem proving requires exact logical derivations. Small mistakes are not tolerated by the proof checker. Early works on neural theorem proving started almost a decade ago (Rocktäschel and Riedel 2017; Bansal et al. 2019; Yang and Deng 2019) and typical theorem proving environments include Lean, Coq, Isabelle, and Mizar. They laid the groundwork for using general-purpose LLMs for theorem proving (Jiang et al. 2022; Yang et al. 2023). Many challenges remain in this direction. For example, high-quality proof corpora are relatively small compared to internet-scale text. Theorem proofs can also be very long and even large models can hallucinate or lose track of intermediate states. See Glazer et al. (2024) for an example of scenarios where LLMs still struggle.

Another direction equips LLMs with code-writing to enhance their problem-solving abilities. The LLM's job is to write correct, logically consistent code, and then a downstream system (e.g., a Python interpreter) executes that code to obtain the result. Gao et al. (2023) is a good example, where the LLM's chain-of-thought is effectively replaced by or augmented with a Python function that solves each sub-step of the problem. Romera-Paredes et al. (2024) is another prime example, where an LLM paired with a systematic evaluator pushed beyond the boundary of human knowledge on the cap set problem and discovered an asymptotic lower bound that was the largest improvement in 20 years. More broadly, this stream of literature is also related to code generation (Chen et al. 2021; Li et al. 2022) and tool-use (Schick et al. 2023) by LLMs.

As mentioned before, the rapid development of LLMs' abilities to solve math problems generally does not consider the aspect of modeling. A series of work, primarily by OR researchers, have discussed successes in marrying LLMs with techniques of formulating optimization problems and solving them (Ramamonjison et al. 2022; Astorga et al. 2025; Xiao et al. 2024; Jiang et al. 2025a; Mostajabdaveh et al. 2024; AhmadiTeshnizi et al. 2024; Huang et al. 2025; Jiang et al. 2025b); valuable datasets and benchmarks have also been published. A few works have also attempted to replicate behavioral hypotheses in OR using LLM agent-based simulations (Kirshner 2024). In contrast, studies on LLMs' abilities to solve stochastic modeling problems have been missing. With this paper, we aim to take the first step towards this direction and hope to inspire future research.

3 HOMEWORK PROBLEMS TEST CASE

3.1 Dataset

To the best of our knowledge, no dataset featuring a list of stochastic modeling problems and solutions is publicly available. Ideally, the dataset should consist of a large number of mathematical models and analytical results from academic papers on stochastic modeling, but as a start we will focus on course settings as a prerequisite. Most textbooks available in digital forms do not have readily available solutions, and there are also copyright issues. Therefore, we manually sourced problems and solutions from related courses at Columbia University. The first version of the dataset has 175 problems and solutions, divided into three categories: probability, stochastic processes, and modeling. However, a quick inspection shows that a large proportion of the problems are educative and not suitable for evaluation; see Example 1 and 2 for two sample problems.

Example 1 (A classical result in probability theory)**.** *Let X be a non-negative random variable with cumulative distribution function F. Show that $\mathbb{E}[X] = \int_0^\infty \bar{F}(x)dx$, where $\bar{F}(x) = 1 - F(x)$.*

Example 2 (A classical result in stochastic processes)**.** *Prove the backwards martingale convergence theorem.*

These problems are meant to educate students about classical results that are important enough but are not covered in class often due to time constraints. It is reasonable to surmise that these classical

problems have solutions that can be readily found online. As a result, it is very likely that LLMs have seen the solutions in their training data. Therefore, we screened the dataset to take out questions of similar nature. The final dataset contains 71 problems and solutions, where 37 are on probability theory, 23 are on stochastic processes, and 11 are on modeling. Below are some examples:

Example 3 (A sample problem in probability theory)**.** *Consider a sequence of i.i.d. random variables $X_1, X_2, \ldots$, each having an exponential distribution with parameter 1. Let $M_n := \max\{X_1, \ldots, X_n\}$. (a) Let $Y_n = X_n \mathbf{1}\{X_n \leq \log n\}$ denote a truncated exponential, i.e., $Y_n = X_n$ if $X_n \leq \log n$ and equals zero otherwise. Prove that $Y_n \neq X_n$, i.o., almost surely. (b) Prove that $M_n / \log n \to 1$ almost surely as $n \to \infty$, where* log *denotes the natural logarithm.*

Example 4 (A sample problem in stochastic processes)**.** *Consider an irreducible discrete-time Markov chain X_n on a finite state space. Let P denote its transition probability matrix. A function f is said to be superharmonic if $f(x) \geq \sum_y P(x,y) f(y)$ or equivalently $f(X_n)$ is a supermartingale. Show that the Markov chain is recurrent if and only if every nonnegative superharmonic function is constant.*

Example 5 (A sample modeling problem)**.** *Consider a queueing system where each job is "fed-back" to the input buffer a finite and random number of times. The input flow to the system $\{(s_n, t_n) : n \in \mathbb{Z}\}$ is assumed to be stationary and ergodic. Provide and prove conditions under which the system has a finite steady-state which the system couples to in finite time.*

Naturally, these problems vary in difficulty and the knowledge required. For instance, Example 3 requires basic knowledge of measure theory; Example 4 requires basic knowledge of Markov chains; Example 5 requires a good understanding of G/G/1 queues and sample-path analyses. Overall, we believe this dataset is representative of the problems that students in a graduate-level stochastic modeling course would encounter.

3.2 LLMs, Prompt Template, and Solving the Problems

We consider 6 LLMs: GPT-4o, o1, o3-mini (OpenAI 2025), Claude 3.5 Sonnet (Anthropic 2024), Llama 3.3 70B Instruct Turbo (Grattafiori et al. 2024), and DeepSeek-R1 (Guo et al. 2025) (we use Together AI (2025) to access Llama and DeepSeek models). While many other LLMs exist and may be of interest, we believe the ones selected are representative of popular and state-of-the-art LLMs. For each LLM and each problem, we call the corresponding API to obtain a solution by the following prompt template:

Prompt Template for HW Problems

You are given a problem from probability theory and stochastic modeling. You need to solve this problem rigorously to the best of your ability. Please provide as many details and explanations as you can. The problem is as follows: \n\n`{problem}`

We use this prompt to evaluate the inherent capabilities of LLMs without additional enhancements. Current LLMs (e.g., o1, o3-mini, DeepSeek-R1) already incorporate reasoning abilities and advanced prompting and sampling techniques. Therefore, we intentionally avoid using complex prompts, chain-of-thought methods (Wei et al. 2022), or specialized sampling approaches in our testing. While we acknowledge that fine-tuning and prompt engineering could potentially improve performance, we reserve these approaches for future exploration.

3.3 Evaluating LLMs' Solutions

Ideally, for a rigorous evaluation of LLMs' solutions, we would have qualified human graders to grade each of LLMs' answers. However, due to budget constraint, a more scalable and valid alternative is the "LLM-as-a-judge" approach (Zheng et al. 2023), where strong LLM models are used to evaluate LLM performance. In our case, we use GPT-4o as the judge. For each LLM's answer, we provide the correct solution and ask the judge to assign a score between 0 and 100 (with 100 meaning the best) to the answer. We also ask the judge to provide comments on the solution and justify the score it gives.

By manually inspecting the judge's scores and comments, we find the evaluations reasonable and sometimes more comprehensive than those that would be provided by a human counterpart. To make the evaluation process more robust, we sample three independent scores from the judge (by calling the API three times) for each problem and take the average for the final score. This procedure mimics pooling scores assigned by three independent human graders. We found that, in general, the three scores do not differ much, underlining the reliability of using an LLM as the judge. In Section 4, we also compare the LLM's score with human scores and show further evidence for alignment.

3.4 Results

The LLMs' average scores (and standard errors in parenthesis) are summarized in Table 1.

Table 1: LLMs' scores on homework problems.

LLM	Probability theory	Stochastic process	Modeling	Total
GPT-4o	81.85 (1.95)	78.72 (2.65)	74.67 (1.57)	79.72 (1.39)
o1	**95.46 (0.47)**	**95.28 (0.56)**	**90.61 (1.96)**	**94.65 (0.48)**
o3-mini	**96.97 (0.36)**	**96.25 (0.54)**	**92.58 (1.29)**	**96.05 (0.37)**
Claude 3.5 Sonnet	88.74 (1.09)	86.96 (1.33)	73.48 (3.74)	85.80 (1.12)
Llama 3.3 70B Instruct Turbo	78.36 (2.47)	75.99 (2.70)	58.18 (3.49)	74.46 (1.85)
DeepSeek-R1	84.30 (2.00)	73.33 (4.44)	64.85 (5.03)	77.73 (2.13)

While we believe that these scores are generally reasonable, we caution against taking these scores to be exact because they are, after all, generated by GPT-4o and thus may have certain bias (Zheng et al. 2023). Nevertheless, several qualitative observations can be made:

1. First and foremost, if we set 60% as the cutoff for passing a stochastic modeling course, all models would pass with flying colors, with o1 and o3-mini being the top performing.
2. Secondly, models with higher scores seem to have smaller standard errors as well. This may be because that they have been optimized to be more accurate and be more aligned with human preferences, which may reduce the variability in their answers.
3. Lastly, for all models, modeling problems seem to be the hardest, perhaps due to their more open-ended nature.

4 QUALIFICATION EXAM PROBLEMS

In light of the observations from the last section, we believe the state-of-the-art LLMs have abilities on par with human PhD students in the field. To make the evaluation more rigorous, we next procure a set of qualification-exam problems to test the LLMs. It is a common tradition in many fields to test a PhD student's ability with a qualification exam after the student has passed all required courses. Only those who pass the exam will be allowed to proceed to the next stage of their PhD and become PhD candidates. Since the LLMs exhibit superior performance in a course-like setting, it would be interesting to see how they perform in a real exam. To ensure fairness, we will manually grade the LLMs' answers; we will also compare our scores with those generated by GPT-4o to provide evidence for the validity of using GPT-4o as a grader.

Results from the last section suggest that o1 and o3-mini have the best performance. Therefore, we will manually grade the answers generated by these two models. As a comparison, we will also consider answers from Claude 3.5 Sonnet since it was the second runner-up. Recall that we observed that modeling problems seem to be the most challenging for these models. By manually inspecting the LLMs' answers,

we are also convinced that the LLMs can answer any probability theory and stochastic process problems well in the exam. Therefore, we will focus on testing these LLMs on modeling problems.

4.1 Dataset

We carefully select 8 modeling problems that have appeared in past qualification exams in the Decision, Risk, and Operations Division at Columbia Business School. A common theme of these problems is to model some real-life problem with an abstract model. The students are asked to analyze the model to arrive at sensible decisions for the real-life problem. Examples include modeling the spread of a virus by a branching process, variants of the newsvendor problem, optimizing assortment planning using Markov chains, and stability conditions for complex queueing systems. In other words, though these problems are still limited to clean models, they have a practical footing and are representative of "analyzable" real-life stochastic modeling problems. Since they are exam problems, they are also less open-ended compared to their counterparts in homework to ensure fair grading. On average, these problems are designed to be challenging for junior PhD students. Every year, a selected committee of faculty would design new qualification exam problems. Since past problems and their solutions are private, it is unlikely that LLMs' training set contains the exact problems.

Due to departmental regulations, we are unable to share actual problems from past qualification exams. To provide a sense of the exam's format and style, we present below a sample modeling problem that is made available to first-year PhD students as part of their preparation. Since its primary purpose is to illustrate the structure and expectations of the exam, the problem is on the easier end of the spectrum compared to typical qualification exam problems.

Example 6 (A sample qualification exam modeling problem)**.** *Consider a non-preemptive FIFO queue with infinite buffer. Requests arrive according to a Poisson process with rate λ, and each has i.i.d. workload $w \sim Exp(\mu)$. The service proceeds as follows. Each request is initially processed for up to θ time units. If completed within θ, it exits the system and the next request (if any) begins service. If not, then the system restarts service in a mode that is divided into two steps: 1) the request is broken into n sub-tasks that are executed in parallel by n servers, where the processing times of the sub-tasks are i.i.d., uniformly distributed in the interval $[0.5w/n, 1.5w/n]$; and 2) the results of all n sub-tasks are combined to complete the service of the original request, in a step that can only commence after all n sub-tasks are completed and its duration is exponentially distributed with rate 2μ, independent of the processing times of the sub-tasks and of the processing requirements of any other requests. (a) What is the stability condition for the system? (b) What is the steady-state expected sojourn time (waiting time + service time) for a request?*

We want to emphasize that the qualification exams at Columbia Business School, especially the stochastic modeling part, are not easy by any means. The core course that prepares students for this exam is widely known to be hard in the community. Students often spend a significant amount of time taking this class and preparing for this exam. Therefore, though we use a small dataset, the evaluation remains meaningful: as educators and researchers in OR, we are genuinely curious about how LLMs perform on these problems, and the results may inform how we teach and assess students in the future.

4.2 Results

We obtain answers from the LLMs in the same fashion as before. We also obtain GPT-4o's grading for these answers. The LLMs' scores (average scores and standard error in parenthesis where applicable) are summarized in Table 2. As evidenced by the scores, though modeling problems seem to be more challenging for LLMs than other types of problems, with 60% being the cutoff for passing, all three models can pass the qualification exam with flying colors. As a reference, though exact score distributions of past qualification exams at Columbia Business School are unknown, across the years scores border-lining the 60% cutoff or lower consistently occurred. In this sense, under the same evaluation framework for humans, these models have demonstrated capabilities at least on par with PhD candidates in the field.

Table 2: LLMs' scores on qualification exam problems.

Problem	**o1**		**o3-mini**		**Claude 3.5 Sonnet**	
	Scores by GPT-4o	Scores by human	Scores by GPT-4o	Scores by human	Scores by GPT-4o	Scores by human
Problem 1	95	94	93.33	94	81	74
Problem 2	95	96	96.67	96	85	88
Problem 3	95	91	95	94	86.67	85
Problem 4	98.33	100	100	100	95	88
Problem 5	96.67	95	96.67	100	91.67	100
Problem 6	95	95	98.33	95	78.33	85
Problem 7	94.67	85	98.33	87	85	80
Problem 8	96.67	100	93.33	100	73.33	77
Total	95.79 (0.43)	94.5 (1.61)	96.46 (0.80)	95.75 (1.47)	84.5 (2.31)	84.63 (2.66)

How well the top LLMs are solving these problems and how fast they are improving (see Table 1) are indeed impressive! But of course, the LLMs' answers are not without flaws. Upon closely examining their answers, we found that, for almost all problems, they have the right intuition and high-level arguments. Where points are taken off, it was mostly because of missing calculations or proof steps. This finding suggests that these LLMs can be helpful assistants for solving stochastic modeling problems, but they tend to generate answers that might lack rigor.

4.3 Grading Alignment

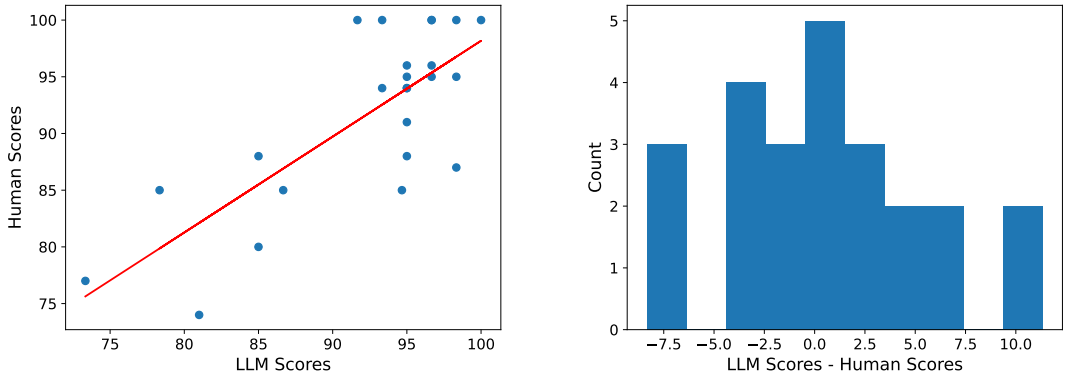
Given scores by humans, it is natural to ask whether the scores given by GPT-4o aligns well with "real" scores. Across the 24 scores, we calculated the Pearson correlation coefficient to be 0.77 (Figure 1, left panel). The distribution of score differences (GPT-4o's scores minus human's scores) is approximately symmetric with a mean of 0.63 and standard error of 1.00 (Figure 1, right panel). The absolute difference has a mean of 3.82 and a maximum of 11.33. Since the maximum score possible is 100, we believe the alignment between GPT-4o's grading and human's is well and the LLM-as-a-judge method is reliable in our context. We also acknowledge that a more comprehensive and large-scale evaluation would be beneficial and leave it for the future work.

5 SIMULATION-OPTIMIZATION PROBLEMS

Having shown that LLMs perform well in a classroom setting, in this section we discuss the ability of different LLMs in solving and implementing simulation-optimization methods on a testbed of problems that more closely resemble those encountered in practice.

5.1 Dataset and Evaluation

We evaluate five LLMs—GPT-4o, o1, o3-mini, Claude 3.5 Sonnet, and DeepSeek-R1—on six optimization problems from the `SimOpt` library, a well-known benchmark suite for noisy simulation optimization in OR (Eckman et al. 2023). Each model is prompted five times with problem descriptions from `SimOpt`. We execute the Python code generated by the models and compare their solutions to those produced by baseline algorithms implemented in `SimOpt`, including RandomSearch, ASTRO-DF, Nelder-Mead, STRONG, SPSA, ADAM, and ALOE (see Dong et al. (2017) for algorithmic details and comparisons). For each problem, we report the objective values achieved by the models and benchmark them against the best solver identified by Dong et al. (2017). These values may differ considerably due to variations



(a) Scores given by GPT-4o vs. given by human.

(b) Distribution of grading differences.

Figure 1: Grading alignment.

in the simulation environments implemented by the different models. Consequently, we also focus on the algorithmic strategy that each model employs by manually analyzing the solutions they propose. Each model solution is allocated the same computational budget, where the budget refers to the number of simulation replications over the entire course of the search for optimal solutions (Eckman et al. 2023). To ensure consistency, we use the same prompt template for all five LLMs. The prompt is crafted by merging the problem descriptions from the `SimOpt` library with a prompt template (omitted due to space constraint), which is then refined using o1 and manually verified for correctness.

5.2 Results

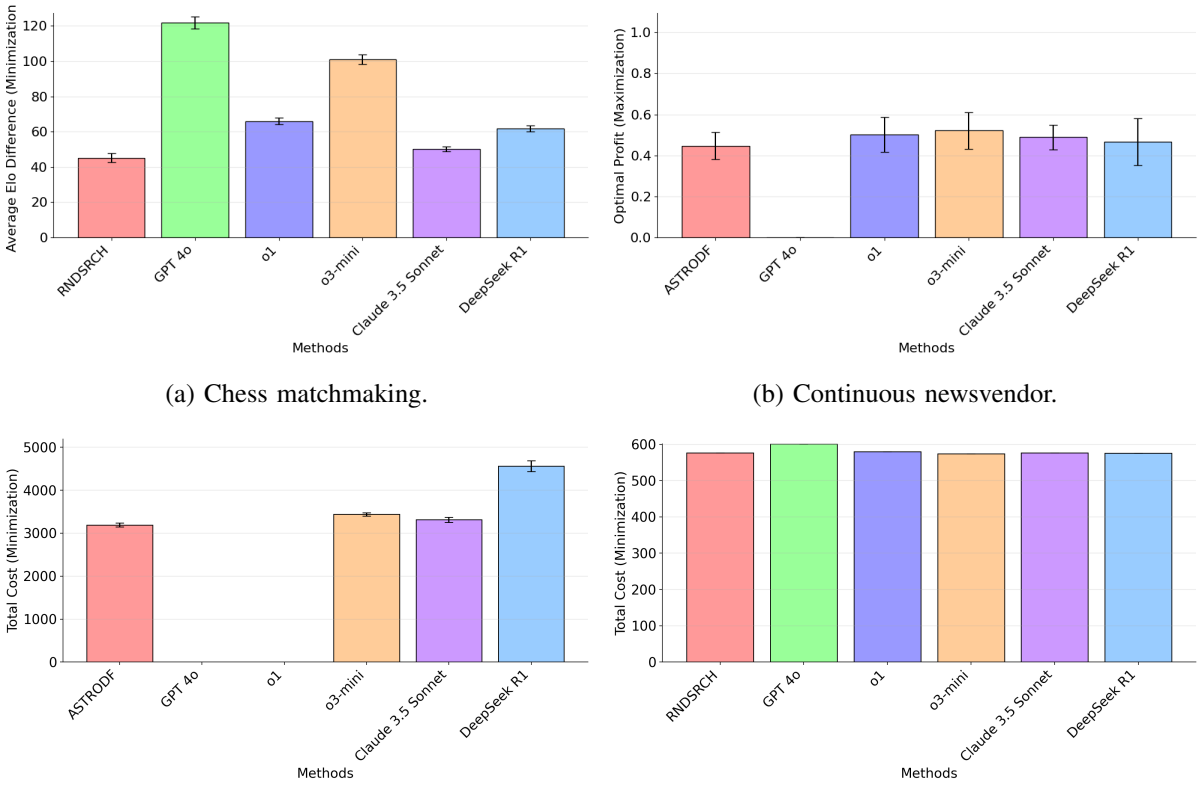
In Figure 2, we showcase the performance of the solution proposed by the different models. If the performance of a model is missing, it means that the model failed to produce reasonable code despite multiple attempts. Below, we summarize our main findings, followed by problem-specific analyses.

1. Claude 3.5 Sonnet delivers the strongest overall performance, achieving near-optimal solutions for five out of six problems. On the other hand, despite their theoretical exam prowess, GPT-4o critically fails on the textbook Continuous Newsvendor problem and o1 can even fail to generate numerical solutions.
2. Consistent methodological preferences surface—Claude leverages binary/differential evolution; DeepSeek-R1 defaults to coordinate descent; and o1 employs domain-constrained grid searches.
3. The IronOre problem highlights implementation limitations, with all models producing incomparable solutions.

As evidenced by comparing LLMs' performance on paper-based exams and on these practical problems, the top performer in one scenario may not be the best in the other. State-of-the-art models like GPT-4o or o1 may even fail to reliably produce numerical solutions. The IronOre problem also suggests that the comparison between models still needs human supervision. Overall, these observations suggest that more work is needed to reliably automate the stochastic modeling pipeline.

5.2.1 Chess Matchmaking (ChessMM)

This problem involves matching players on an online chess platform to minimize the average Elo difference between matched pairs, while ensuring that the average waiting time does not exceed a specified threshold ($\delta = 5.0$). In this setting, players arrive according to a Poisson process and their Elo ratings are sampled



(a) Chess matchmaking.

(b) Continuous newsvendor.

(c) Dual sourcing.

(d) Facility sizing.

Figure 2: Performance of different LLMs for simulation-optimization problems. We omit IronOre due to differences in how LLMs implement the simulation environment, making the results incomparable. We also exclude ParamEsti, as all LLMs converge to the optimal solution.

from a truncated normal distribution over the interval $[0, 2400]$. No theoretical optimal solution is known, and `SimOpt`'s best-performing algorithm achieved an average Elo difference of 45.1246. All evaluations are conducted with a fixed budget of 1000 function calls. The different LLM approaches primarily differ in their search strategies for determining the optimal allowable Elo difference threshold, x. For example, GPT-4o's solution implements a line search over the full range $[0, 2400]$. In contrast, o1's approach confines its grid search to the interval $[0, 300]$, effectively focusing its limited budget on a more promising subset of the parameter space, which results in better performance relative to the other methods. Similarly, o3-mini's approach uses grid search but over a wider interval $[10, 2400]$, which can lead to a less concentrated search and suboptimal tuning. Meanwhile, Claude's solution employs a binary search over $[0, 2400]$, efficiently honing in on the smallest threshold that meets the waiting time constraint, while DeepSeek also utilizes grid search over the full range. These differences in search range and methodology underscore how a more targeted exploration—such as the one adopted by o1—can yield superior performance when the computational budget is fixed.

5.2.2 Continuous Newsvendor (CntNv)

This problem considers a vendor who orders a fixed quantity of liquid at the beginning of the day. The liquid is sold to customers at a per-unit price and any unsold inventory is salvaged at a lower per-unit price, while the vendor incurs a per-unit ordering cost. In this classic formulation the optimal order quantity is known in closed form. In our evaluation, we compare ASTRO-DF alongside LLM-derived solutions, all

using a fixed budget of 1000 function evaluations. Among the LLM solutions, GPT-4o failed to produce a reasonable solution, whereas o1 and o3-mini both leveraged the closed-form solution to guide their search. Specifically, o1 combined the closed-form reasoning with a random search strategy, while o3-mini employed a grid search over candidate order quantities. In contrast, Claude utilized a Nelder-Mead algorithm to iteratively converge to the optimum, and DeepSeek applied a grid search approach.

5.2.3 Dual Sourcing (DualSourcing)

This problem requires choosing optimal order-up-to levels for two procurement channels—one regular (lower cost, longer lead time) and one expedited (higher cost, shorter lead time)—so as to minimize the total expected daily cost (comprising holding, penalty, and ordering costs) over n periods. All the LLMs implemented grid search. Overall, the differences in performance across these models can be traced to how they allocate their fixed evaluation budget. o3-mini, Claude, and DeepSeek restrict the search to a plausible, domain-informed range and appropriately replicate the simulations. In contrast, GPT-4o and o1 misinterpreted and incorrectly implemented key aspects of the inventory dynamics—the correct handling of order arrivals, backorders, and the inventory pipeline—leading them to absurd cost estimates.

5.2.4 Facility Sizing (FacSize)

This problem requires selecting nonnegative capacities x_i for three facilities to minimize the total installation cost while ensuring that the probability of a stockout (i.e., at least one facility experiencing demand exceeding its capacity) remains below a certain level. GPT-4o uses a continuous optimization approach with `scipy.optimize.minimize`, simulating demand samples and incorporating the stockout probability as an inequality constraint; however, its estimation of the constraint is imprecise, leading to suboptimal solutions. In contrast, o1 employs a grid search over a discrete range of candidate capacities and reuses pre-generated demand samples to estimate the stockout probability for each candidate, ultimately selecting the lowest-cost solution that meets the risk constraint. o3-mini takes a different approach by parameterizing the capacities as $x = \mu + z\sigma$ and performing a binary search on the scalar parameter z to efficiently identify the smallest safety margin that satisfies the stockout constraint. Claude also relies on an iterative binary search, starting with bounds based on the mean and standard deviations of the demand distribution, and refines these bounds through repeated simulations to robustly estimate the stockout probability, thereby finding a cost-effective solution. DeepSeek initially derives a solution by setting baseline capacities based on normal quantiles and then applies coordinate descent to iteratively lower each capacity while ensuring that the overall stockout probability remains within acceptable limits.

5.2.5 Iron Ore Production with Exogenous Stochastic Price (IronOre)

This problem models a mine producing and selling an item (such as iron ore) on a spot market where the daily price P_t follows a truncated mean-reverting random walk. Every day the decision maker observes P_t and the current inventory level, then makes production and sales decisions according to four thresholds: x_1 is the price above which production starts or continues, x_2 is the inventory level above which production is halted, x_3 is the price below which production is stopped, and x_4 is the price above which the entire inventory is sold. Production is limited to a maximum daily amount and capacity constraints, while holding costs apply to unsold inventory. Among the SimOpt algorithms, ASTRO-DF performed the best in terms of maximizing profit. All the LLMs differ in the way they implement the simulation environment for this problem, so their numerical results are not directly comparable. Therefore, we describe only qualitatively the strategies each one uses: GPT-4o performs a random search over the continuous threshold space using `scipy.optimize.minimize`; o1 uses a grid search over a broad range of threshold candidates, applying pre-generated price paths to evaluate each policy, but its ordering of decisions appears inconsistent; o3-mini parameterizes the thresholds relative to the mean and standard deviations of the price process and uses a binary search on a safety margin parameter combined with multiple replications to refine the

candidate solution; Claude employs differential evolution, running multiple simulation trials per evaluation to iteratively improve the threshold values; and DeepSeek initiates its search with a Bonferroni-based heuristic and then applies coordinate descent to iteratively adjust the thresholds.

5.2.6 Parameter Estimation (ParamEsti)

This is a classical problem in which the objective is to recover the unknown parameter vector x^* of a two-dimensional gamma distribution from i.i.d. observations. Both SimOpt algorithms and those implemented by the various LLMs converge to the optimal solution. We omit their implementation details for brevity.

6 CONCLUSIONS

In this work, we evaluated LLMs' abilities to solve stochastic modeling problems through a series of homework problems, qualification-exam problems, and simulation-optimization problems. Our findings suggest that these models have the potential to automate the stochastic modeling pipeline at a level comparable with human experts, and we hope this work will inspire future research in developing intelligent OR agents that can help make real-world decisions reliably and at scale.

REFERENCES

AhmadiTeshnizi, A., W. Gao, H. Brunborg, S. Talaei, and M. Udell. 2024. "OptiMUS: Scalable Optimization Modeling with (MI)LP Solvers and Large Language Models". In *Proceedings of the 41st International Conference on Machine Learning*. July 21st-24th, Vienna, Austria, 577–596.

Anthropic 2024. "Claude 3.5 Sonnet". https://www.anthropic.com/news/claude-3-5-sonnet. Accessed 14th August 2025.

Astorga, N., T. Liu, Y. Xiao, and M. van der Schaar. 2025. "Autoformulation of Mathematical Optimization Models Using LLMs". In *Proceedings of the 42nd International Conference on Machine Learning*. July 13th-19th, Vancouver, Canada.

Bansal, K., S. Loos, M. Rabe, C. Szegedy, and S. Wilcox. 2019. "HOList: An Environment for Machine Learning of Higher Order Logic Theorem Proving". In *Proceedings of the 36th International Conference on Machine Learning*. June 9th-15th, Long Beach, CA, US, 454–463.

Brown, T., B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, *et al*. 2020. "Language Models are Few-Shot Learners". In *Proceedings of the 34th Conference on Neural Information Processing Systems*. December 6th-12th, Virtual, 1877–1901.

Chen, M., J. Tworek, H. Jun, Q. Yuan, H. P. d. O. Pinto, J. Kaplan, *et al*. 2021. "Evaluating Large Language Models Trained on Code". *arXiv preprint arXiv: 2107.03374*.

Cobbe, K., V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, *et al*. 2021. "Training Verifiers to Solve Math Word Problems". *arXiv preprint arXiv: 2110.14168*.

Dai, J. G., and J. M. Harrison. 2020. *Processing Networks: Fluid Models and Stability*. 1st ed. Cambridge: Cambridge University Press.

Dong, N. A., D. J. Eckman, X. Zhao, S. G. Henderson, and M. Poloczek. 2017. "Empirically Comparing the Finite-time Performance of Simulation-optimization Algorithms". In *2017 Winter Simulation Conference (WSC)*, 2206–2217 https://doi.org/10.1109/WSC.2017.8247952.

Eckman, D. J., S. G. Henderson, S. Shashaani, and R. Pasupathy. 2023. "SimOpt: A Testbed for Simulation-Optimization Experiments". *INFORMS Journal on Computing* 35(2):495–508.

Gao, L., A. Madaan, S. Zhou, U. Alon, P. Liu, Y. Yang, *et al*. 2023. "PAL: Program-aided Language Models". In *Proceedings of the 40th International Conference on Machine Learning*. July 23rd-29th, Honolulu, HI, US, 10764–10799.

Glazer, E., E. Erdil, T. Besiroglu, D. Chicharro, E. Chen, A. Gunning, *et al*. 2024. "FrontierMath: A Benchmark for Evaluating Advanced Mathematical Reasoning in AI". *arXiv preprint arXiv: 2411.04872*.

Grattafiori, A., A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, *et al*. 2024. "The Llama 3 Herd of Models". *arXiv preprint arXiv: 2407.21783*.

Guo, D., D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, *et al*. 2025. "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning". *arXiv preprint arXiv: 2501.12948*.

Hendrycks, D., C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, *et al*. 2021. "Measuring Mathematical Problem Solving With the MATH Dataset". In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*. December 6th-14th, Virtual.

Huang, C., Z. Tang, S. Hu, R. Jiang, X. Zheng, D. Ge, *et al*. 2025. "ORLM: A Customizable Framework in Training Large Models for Automated Optimization Modeling". *arXiv preprint arXiv: 2405.17743*.

Jiang, A. Q., W. Li, S. Tworkowski, K. Czechowski, T. Odrzygóźdź, P. Mił oś, *et al*. 2022. "Thor: Wielding Hammers to Integrate Language Models and Automated Theorem Provers". In *Proceedings of the 36th Conference on Neural Information Processing Systems*. November 28th-December 9th, New Orleans, LA, US, 8360–8373.

Jiang, C., X. Shu, H. Qian, X. Lu, J. Zhou, A. Zhou *et al*. 2025a. "LLMOPT: Learning to Define and Solve General Optimization Problems from Scratch". In *The Thirteenth International Conference on Learning Representations*. April 24th-28th, Singapore.

Jiang, C., X. Shu, H. Qian, X. Lu, J. Zhou, A. Zhou *et al*. 2025b. "LLMOPT: Learning to Define and Solve General Optimization Problems from Scratch". In *The Thirteenth International Conference on Learning Representations*. April 24th-28th, Singapore.

Kirshner, S. 2024. "Artificial Agents in Operations Management Experiments". *SSRN 4726933*.

Li, Y., D. Choi, J. Chung, N. Kushman, J. Schrittwieser, R. Leblond, *et al*. 2022. "Competition-Level Code Generation with AlphaCode". *Science* 378(6624):1092–1097.

Mostajabdaveh, M., T. T. Yu, R. Ramamonjison, G. Carenini, Z. Zhou, and Y. Zhang. 2024. "Optimization Modeling and Verification from Problem Specifications Using a Multi-Agent Multi-Stage LLM Framework". *INFOR: Information Systems and Operational Research* 62(4):599–617.

OpenAI 2025. "OpenAI Platform Models". https://platform.openai.com/docs/models. Accessed 14th August 2025.

Ramamonjison, R., H. Li, T. Yu, S. He, V. Rengan, A. Banitalebi-dehkordi, *et al*. 2022. "Augmenting Operations Research with Auto-Formulation of Optimization Models From Problem Descriptions". In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track*. December 7th-11th, Abu Dhabi, UAE, 29–62.

Rocktäschel, T., and S. Riedel. 2017. "End-to-end Differentiable Proving". In *Proceedings of the 31st Conference on Neural Information Processing Systems*. December 4th-9th, Long Beach, CA, US, 3791–3803.

Romera-Paredes, B., M. Barekatain, A. Novikov, M. Balog, M. P. Kumar, E. Dupont, *et al*. 2024. "Mathematical Discoveries from Program Search with Large Language Models". *Nature* 625(7995):468–475.

Schick, T., J. Dwivedi-Yu, R. Dessi, R. Raileanu, M. Lomeli, E. Hambro, *et al*. 2023. "Toolformer: Language Models Can Teach Themselves to Use Tools". In *Proceedings of the 37th Conference on Neural Information Processing Systems*. December 10th-16th, New Orleans, LA, US, 68539–68551.

Together AI 2025. "Together AI - The AI Acceleration Cloud". https://www.together.ai/. Accessed 14th August 2025.

Wei, J., X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, *et al*. 2022. "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models". In *Proceedings of the 36th Conference on Neural Information Processing Systems*. November 28th-December 9th, New Orleans, LA, US, 24824–24837.

Xiao, Z., D. Zhang, Y. Wu, L. Xu, Y. J. Wang, X. Han, *et al*. 2024. "Chain-of-Experts: When LLMs Meet Complex Operations Research Problems". In *The Twelfth International Conference on Learning Representations*. May 7th-11th, Vienna, Austria.

Yang, K., and J. Deng. 2019. "Learning to Prove Theorems via Interacting with Proof Assistants". In *Proceedings of the 36th International Conference on Machine Learning*. June 9th-15th, Long Beach, CA, US, 6984–6994.

Yang, K., A. Swope, A. Gu, R. Chalamala, P. Song, S. Yu, *et al*. 2023. "LeanDojo: Theorem Proving with Retrieval-Augmented Language Models". In *Proceedings of the 37th Conference on Neural Information Processing Systems*. December 10th-16th, New Orleans, LA, US, 21573–21612.

Zheng, L., W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, *et al*. 2023. "Judging LLM-as-a-judge with MT-bench and Chatbot Arena". In *Proceedings of the 37th Conference on Neural Information Processing Systems*. December 10th-16th, New Orleans, LA, US, 46595–46623.

AUTHOR BIOGRAPHIES

AKSHIT KUMAR is a PhD student in the Decision, Risk, and Operations division at Columbia Business School, working on problems in dynamic resource allocation and online matching. His email address is ak4599@columbia.edu.

TIANYI PENG is an Assistant Professor in the Decision, Risk, and Operations division at Columbia Business School and a member of the Data Science Institute at Columbia University. His research focuses on AI for decision-making systems. His email address is tianyi.peng@columbia.edu.

YUHANG WU is a PhD student in the Decision, Risk, and Operations division at Columbia Business School. He's interested in multi-agent systems and agent-based simulation. His email address is yuhang.wu@columbia.edu.

ASSAF ZEEVI is the Kravis Professor of Business in the Decision, Risk, and Operations division at Columbia Business School, and at the Data Science Institute. His research focuses on the intersection of operations research, machine learning and data science. His email address is assaf@gsb.columbia.edu