Enhancing LLMs for Power System Simulations: A Feedback-driven Multi-agent Framework

Response Letter

Mengshuo Jia[‡], Zeyu Cui[⋆], Gabriela Hug[‡]

[‡]*Power Systems Laboratory, ETH Zurich, 8092 Zurich, Switzerland*

[⋆]*Qwen Team at DAMO Academy, Alibaba Group, 311121 HangZhou, China*

In order to clearly distinguish the contents in this response letter, the comments of the editor and reviewers are provided in **blue**, the point-to-point responses are given in **black**, and the revised portions of the manuscript included here and in the revised paper are both in **red**.

Please also note that, the dotted-line hyperlinks in this response letter (e.g., **Response to Comment 1.1**) are designed as clickable links for direct navigation to the corresponding responses. However, the submission system's conversion process may disable these interactive features. For convenience, an identical version of this letter with active navigation is available by clicking here.

Response to the Editor

Editor's Comments to the Authors: Although some comments have been addressed in the revised manuscript, a reviewer raised several concerns, i.e., the paper lacks discussion on technical challenges and generalizability beyond power systems; detailed analysis of feedback loops and stopping criteria is missing; and the scoring method, task categorizations, and scheme notations are not clearly explained. Minor issues include unclear figures and grammatical errors. Authors are suggested to revise their paper seriously according to the comments of the reviewer.

***Response to the Editor**:* We thank the Editor for providing us with the opportunity to further improve the manuscript. We appreciate the clear articulation of key concerns, which have guided us in making substantial and targeted revisions. In response, we have carefully and seriously revised the manuscript to address all of the issues raised, focusing on the following aspects:

(1) Addressing technical challenges and generalizability:

- We have now systematically discussed the technical challenges addressed by this work, and clarified the inherent capabilities of LLMs, as well as the domain-specific strategies needed for power system simulations. Furthermore, we have provided new discussions on the extensibility of the proposed framework, emphasizing its potential to be applied beyond simulation tasks. Please refer to **Response to Comment 1.1** for details.

(2) Providing additional evaluations of feedback loops and stopping criteria:

- We have incorporated extensive new experimental results and analyses to examine how tasks evolve toward successful completion or termination by the stopping criterion. These results cover both the DALINE and MATPOWER environments under varying task complexities. In addition, we employed an increased attempt budget to provide deeper insights into feedback loop behaviors. For details, please refer to **Response to Comment 1.2**.

(3) Clarifying the scoring method, task categorization, and scheme notations:

- We have refined the description of the scoring methodology and provided specific examples to clarify what constitutes irrelevant settings. In addition, we have explained that scoring was conducted manually by human experts and clarified the potential for future automated evaluation. For details, please refer to **Response to Comment 1.3**.
- We have also improved the clarity of scheme notations in the relevant table and figure captions to ensure clearer and more consistent references throughout the manuscript. These revisions are detailed in **Response to Comment 1.4**.

(4) Improving figure clarity and correcting grammatical issues:

- We have provided detailed explanations to clarify the definitions and relationships among "All", "All-Complex", "All-Standard", "First Attempt", and "Final Attempt". Please refer to **Response to Comment 1.5** for details.
- We have also addressed additional issues by adding remark indexes, correcting grammatical errors, and thoroughly proofreading the manuscript to further improve the overall writing quality, as discussed in **Response to Comment 1.6**.

Response to Reviewer 1

General Comments to the Author: This paper designed a framework to leverage LLMs for power system simulations. Although the key techniques stem from LLM research in computer science, the authors make efforts to adapt these techniques in the power system simulation fields. The experiment results also seem to demonstrate the roles of each component in the framework. My detailed comments are as follows.

Response to the General Comment: We thank the reviewer for the recognition and insightful suggestions. The reviewer's comments have been instrumental in refining our work, offering critical perspectives that have significantly improved the clarity, structure, and depth of the manuscript. In response, we have made substantial revisions to various parts of the paper. For detailed responses and corresponding revisions, please refer to:

- **Response to Comment 1.1** for a detailed discussion of the technical challenges, inherent LLM capabilities, challenges beyond LLMs' general abilities, and the extensibility of the proposed framework.
- **Response to Comment 1.2** for the addition of extensive new evaluation results and analyses on feedback loop distributions for task completion or termination across different simulation environments. An increased attempt budget has also been adopted in the new evaluations, considering varying task difficulty levels characterized by the number of sub-queries.
- **Response to Comment 1.3** for the refinement of the scoring methodology, clarification of irrelevant settings with specific examples, and explanation of the evaluation procedure.
- **Response to Comment 1.4** for the improvement of scheme notation clarity and consistent referencing throughout the manuscript.
- **Response to Comment 1.5** for the clarification of the meanings and relationships among the figure labels "All", "All-Complex", "All-Standard", "First Attempt", and "Final Attempt".
- **Response to Comment 1.6** for the addition of remark indexes, correction of grammatical issues, and overall proofreading to enhance writing quality.

The detailed point-by-point responses are provided below.

Comment 1.1: Apart from the application side, the authors may discuss the technique challenges of this paper, as well as how the designed framework can be extended beyond the field of power system simulations. This can help the readers understand what components are inherent abilities of LLMs and what are specific to the power system simulations.

Response to Comment 1.1*:* We completely agree that such a discussion can help readers better understand the inherent capabilities of LLMs and the specific designs of the proposed framework for power system simulations. Below, we first discuss the technical challenges addressed in this work, followed by a discussion on the extensibility of the proposed framework. Finally, we provide the revisions made to the manuscript.

(i) Technical challenges of this work

LLMs possess several inherent capabilities that provide a strong foundation for power system simulations. These include robust general language understanding and generation abilities, which enable LLMs to interpret user requests and produce syntactically valid code. In addition, LLMs can process natural language execution feedback (e.g., error messages), offering the potential for iterative refinement. Together, these capabilities make LLMs a promising basis for building agents capable of handling simulation tasks through natural language interaction.

However, applying LLMs to power system simulations presents significant technical challenges that go beyond these general abilities. As discussed in Section I and elaborated throughout Sections II–IV, these challenges include:

- Power system knowledge appears infrequently in LLM training datasets, particularly in complex simulation cases, which limits the models' ability to generalize to specialized tasks.
- Detailed, annotated coding data for power system simulations is scarce, making it difficult for LLMs to fully understand and operationalize simulation procedures.
- Complex simulation tasks often require reasoning across multiple sub-queries and maintaining precise coordination, which remains challenging for LLMs.
- Identifying simulation parameters, functions, and their logical relationships places heavy demands on LLMs. Without adequate domain knowledge, semantic drift during code generation may lead to incorrect implementations.

To overcome these challenges, domain-specific solutions are required. Accordingly, we have proposed an enhanced retrieval module, advanced reasoning mechanisms, and iterative error-feedback strategies. These designs are grounded in power system simulation expertise, including constructing simulation documents, designing chain-of-thought prompts and few-shot examples for reasoning and query planning of simulation tasks, and systematically coordinating all modules into an effective feedback loop. As verified in the case studies, the proposed solutions collectively bridge the gap between general-purpose LLM capabilities and the complex requirements of power system simulations.

(ii) Extensibility of the proposed framework:

While this study focuses on simulation tasks, the proposed framework is readily extensible to broader power systems research applications. Since the adaptability of each module has already been analyzed in Sections II.C, III.D, and IV.D, we discuss the extensibility of the framework here from two complementary perspectives: its underlying principle and its envisioned role.

- **Underlying principle**: At its core, this work demonstrates how LLMs can be empowered to operate effectively in domains where they lack sufficient prior knowledge or high-quality training data. For example, DALINE represents a case where LLMs have no pre-existing knowledge, while MATPOWER reflects a domain with only relatively limited knowledge available. Through the proposed feedback-driven multi-agent framework, LLMs can progressively adapt to these unfamiliar tasks by leveraging retrieval, reasoning, and error-feedback loops. This approach can thus be extended beyond simulations to help LLMs perform reliably in other specialized and knowledge-sparse engineering domains. Within power systems, for example, similar methods can support general power system analysis tasks or assist researchers in designing and validating algorithms for specific power system problems, once the LLMs are adapted to the corresponding tasks.
- **Envisioned role**: Beyond the technical implementation of simulations, this work serves as a bridge to higher-level applications and research workflows. On one hand, the framework can assist human researchers by automating labor-intensive simulation implementation, allowing researchers to focus on conceptual and idea-driven activities. On the other hand, it functions as a natural language interface that can integrate simulations with upstream and downstream power system tasks, helping to bridge heterogeneous data formats and models. More broadly, the simulation-enhanced LLM system can be envisioned as a foundational component in future human-machine collaborative research, where LLMs not only execute simulations but also assist in designing, validating, and refining research ideas through their simulation capabilities. As such, the framework has the potential to evolve into a general-purpose foundation for LLM-based research assistants, applicable both within and beyond the power systems domain.

(iii) Revisions made to the manuscript:

As the challenges and the adaptability of each module have already been discussed in Sections I, II.C, III.D, and IV.D, the revised manuscript further supplements the discussion with the following complementary perspectives:

Section VI. Conclusion

...

Overall, the effectiveness of this work relies on the coordinated integration of general-purpose LLM capabilities (such as natural language understanding and code generation) with carefully designed domain-specific strategies tailored to power system simulations. These strategies are essential for addressing the unique challenges posed by simulation tasks. While this work focuses on simulations, the proposed framework is inherently designed to enable LLMs to adapt to unfamiliar domains and is readily extensible to broader power system research. Furthermore, the proposed simulation-enhanced LLM system may serve as a key component in future human-machine collaborative research, supporting not only simulation execution but also the validation and refinement of research ideas.

...

Comment 1.2: Based on the statements in Section IV-C, the error report and feedback continue until the code meets all requirements or the stopping criterion is reached. For the failure case, I would assume that the reason is the stopping criterion is reached after multiple loops of error reports and feedback. It would be interesting to provide distributions of how many loops need to be experienced until the requirements are met or the stopping criterion is encountered, respectively. The results may be discussed for different difficulty levels of tasks.

Response to Comment 1.2: We fully agree that providing the distributions of feedback loops until task completion or termination is valuable for a better understanding of the process. In response, we have substantially revised the manuscript and supplemented the following simulation results: (i) the distributions of feedback loops required for successful task completion in complex and standard tasks within the DALINE environment; (ii) the corresponding distributions for the MATPOWER environment; and (iii) an extended analysis with the attempt budget increased to 50, to further examine task progression and explicitly capture how many loops are needed until either successful completion or termination by the stopping criterion. Detailed revisions and newly added discussions are as follows:

Section V.B. Evaluation on DALINE → 5. Distribution of Attempts for Task Completion

To further examine the feedback mechanism, Fig. 9 presents the distribution of attempts (feedback loops) required to successfully solve complex and standard tasks. For complex tasks, baseline schemes (e.g., `GPT4o-Sole`) frequently fail or require multiple attempts, indicating that resolving complicated queries (i.e., tasks with more sub-queries and dependencies) poses significant challenges. In contrast, `GPT4o-Full` performs markedly better, with most tasks completed within one or two attempts. For standard tasks, all schemes achieve improved results, and failures are infrequent. `GPT4o-Full` again shows clear advantages, solving the majority of standard tasks in a single attempt. Even less-equipped schemes perform relatively well on standard tasks, suggesting that these tasks impose lower demands on reasoning, retrieval, and error correction, and are thus more manageable for LLMs.

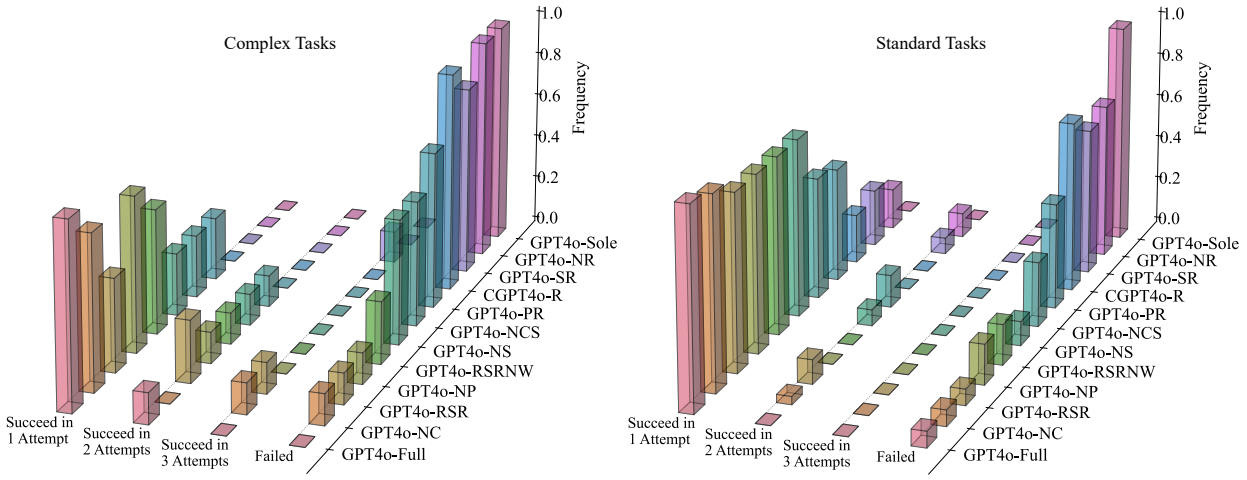

Fig. 9: Distribution of feedback loops (attempts) required for successful task completion in complex (left) and standard (right) tasks. (simulation environment: DALINE; GPT4o version: gpt-4o-2024-05-13; open-source repository for all tasks and results: [link]).

...

Section V.C. Evaluation on DALINE → 3. Distribution of Attempts for Task Completion

Compared to Fig. 9, a similar pattern of the attempt distribution is observed under the MATPOWER simulation environment, as shown in Fig. 13. `GPT4o-Full` still maintains a clear advantage by completing the majority of tasks in the first or second attempt. These results further confirm the effectiveness and generalizability of the proposed framework across different simulation environments.

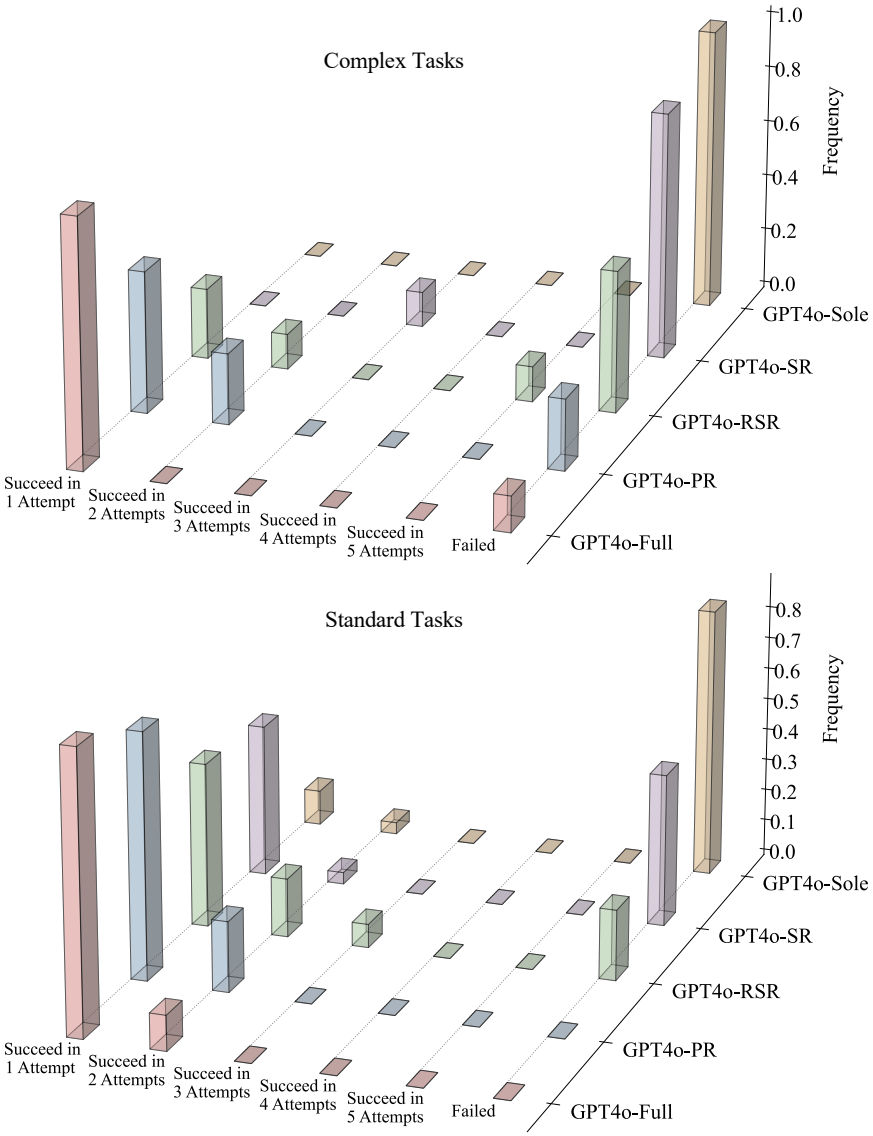

Fig. 13: Distribution of feedback loops (attempts) required for successful task completion in complex (upper) and standard (lower) tasks. (simulation environment: MATPOWER; GPT4o version: gpt-4o-2024-05-13; `o1p-Sole` is only tested by complex tasks; open-source repository for all tasks and results: [link]).

...

Section V.E. Evaluation under Increased Attempt Budget

To further analyze how tasks evolve toward either successful completion or termination by the stopping criterion, Fig. 16 presents the scores achieved in individual attempts with an extended attempt budget of

up to 50. Meanwhile, task difficulty is explicitly characterized by the number of sub-queries within each task. This analysis offers a more comprehensive view of how different schemes perform under varying task complexities, especially when granted extensive opportunities for error correction.

The results in Fig. 16 reveal a clear distinction between schemes and task difficulty levels. Baseline methods struggle significantly with complex tasks — they often either reach the maximum iteration limit (i.e., the stopping criterion) or terminate without execution errors but still fail to satisfy all sub-queries. Typical failure reasons include: (i) invalid option names, which prevent proper execution and remain unresolved even after 50 feedback loops (e.g., `GPT4o-Sole-SFT` in the 2nd and 5th tasks, `GPT4o-Sole` in the 5th task, and `GPT4o-SR` in the 5th task); (ii) correct option names with invalid parameter values produce incorrect but executable code (e.g., `GPT4o-Sole` in the 2nd to 4th tasks and `GPT4o-SR` in the 3rd task); (iii) missing critical option settings, which result in early termination without errors but produce incorrect outputs (e.g., `GPT4o-PR` in the 5th task).

These observations confirm that repeated error-feedback loops alone are insufficient for convergence on challenging tasks without strong reasoning and retrieval capabilities. In contrast, `GPT4o-Full` exhibits robust performance across all difficulty levels, completing all tasks — including those with numerous sub-queries — within only a few attempts. Notably, `GPT4o-Full` only failed at the first attempt for the 5th task, where it misused an option name. This error, however, was successfully corrected in the second attempt.

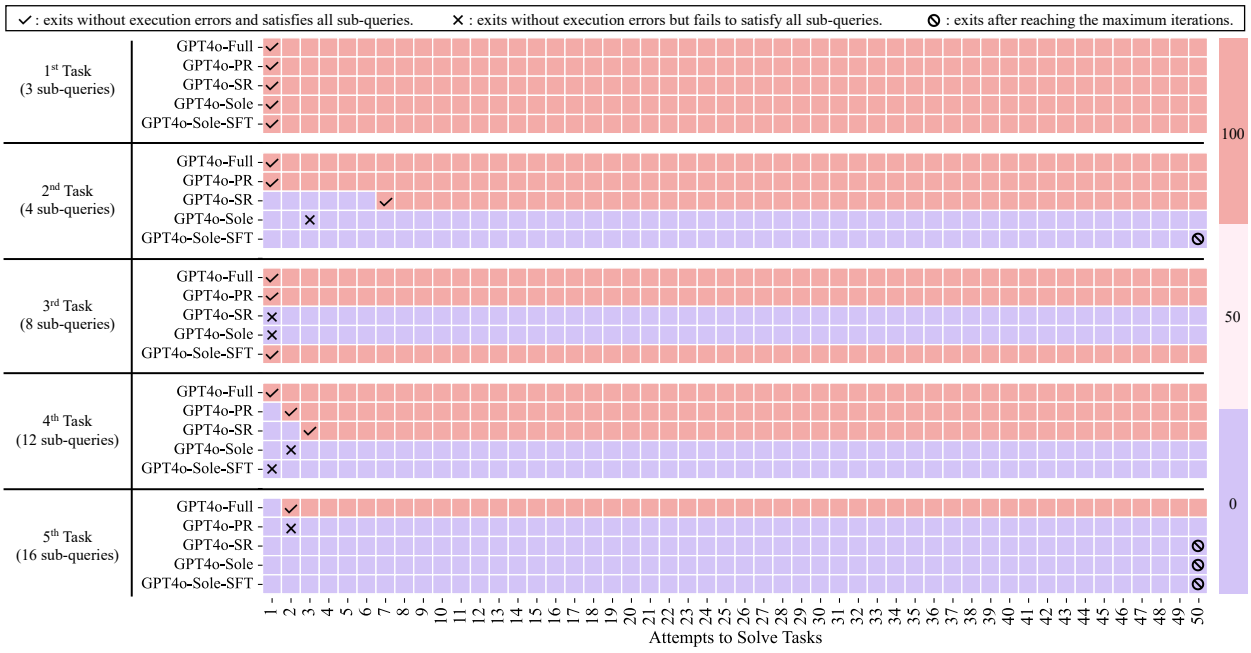

Fig. 16: Scores achieved by each evaluated scheme in individual attempts when handling tasks with different difficulty levels; the bar on the right represents the score scale, where 100, 50, and 0 indicate different score values (simulation environment: MATPOWER; GPT4o version: gpt-4o-2024-08-06; open-source repository for all tasks and results: [link]).

Comment 1.3: When introducing the scoring methods, the authors noted that the point is 50 if the simulation result is correct but contains irrelevant settings. Could you explain (maybe by providing some examples) what cases can be categorized as this? In addition, are the categorizations completed by a human or automatically?

Response to Comment 1.3*:* In response to this comment, we have refined the description of the scoring method, particularly regarding the explanation of irrelevant settings, and added examples to improve clarity and readability, as detailed below:

Section V.A. Settings

...

- $P_{t,i} = 100$ points if all requirements are fully satisfied and no irrelevant settings appear in the code.
- $P_{t,i} = 50$ points if all requirements are satisfied, but the code includes irrelevant or unnecessary settings that do not affect task completion.
- $P_{t,i} = 0$ points if any requirement is not satisfied.

...

Remark 3: To further clarify what constitutes the aforementioned irrelevant settings, preliminary examples are provided here for illustration, with full details to be shown in the subsequent case studies. These include: (i) explicitly setting parameters to their default values when not required (e.g., in DALINE, setting `data.baseType` as `TimeSeriesRand` in Standard Task 2, and `data.program` as `acpf` in Standard Task 7, both by `GPT4o-PR`); and (ii) including unnecessary function calls, such as using `define_constants` in MATPOWER when no constants are actually used (e.g., Standard Task 27 by `GPT4o-Sole`). Such settings reflect imperfections in code generation and therefore receive partial credit.

Regarding implementation, since the proposed framework achieves a high success rate, it indeed demonstrates strong potential for automatically evaluating and scoring the outcomes of different schemes. **However, all categorization and scoring in this study were still conducted manually by human experts** to ensure rigorous and error-free assessment. Nevertheless, supported by the promising performance of the proposed framework, we envision developing a retrieval-supported, fine-tuned LLM-based code-checking agent in the future, to assist human evaluators with categorization, scoring, and judgment tasks, as outlined in the future work section of the Conclusion. Related clarifications have been added to the revised manuscript.

Section V.A. Settings

...

In addition, the scoring was conducted manually by human experts during the evaluation process.

...

Last but not least, all code generation results, along with detailed categorization information (i.e., annotations are provided for each code script generated by the agents, explaining the reasons for success, failure, or the presence of irrelevant parameters), are available in this open-source repository.

Comment 1.4: I would suggest providing some explanations of the evaluated scheme notations in Figs. 7 and 8 so that the readers can understand them better.

Response to Comment 1.4: To address this comment, we have revised the manuscript to improve the clarity of scheme notations. Specifically, we have enhanced the caption of Table I to better explain the evaluated schemes and their corresponding notations. Accordingly, we have updated the caption of Fig. 7 to explicitly reference Table I for definitions of scheme notations. Meanwhile, we have clarified that the same notations apply throughout the manuscript. Detailed revisions are presented below:

TABLE I: Evaluated schemes by distinct combinations of the proposed strategies; a checkmark indicates inclusion of the corresponding strategy (GPT4o: APIs of gpt-4o-2024-05-13 and gpt-4o-2024-08-06, with the exact version specified in each evaluation; CGPT4o: ChatGPT4o; o1p: o1-preview).

	GPT4o Full	GPT4o PR	GPT4o RSR	GPT4o SR	GPT4o Sole	GPT4o NC	GPT4o NP	GPT4o NS	GPT4o NR	GPT4o NCS	GPT4o RSRNW	CGPT4o R	o1p Sole	GPT4o Sole-SFT
Query Planning	✓	✓				✓	✓	✓		✓	✓			
Triple-based Structured Option Document	✓	✓	✓	✓		✓		✓		✓	✓	✓		
Chain of Thought Prompting	✓		✓				✓	✓	✓		✓			
Few-Shot Prompting	✓		✓			✓	✓		✓		✓			
Static Basic Knowledge	✓		✓			✓	✓	✓	✓	✓	✓			
Environmental Acting and Feedback	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Proposed RAG	✓	✓				✓	✓	✓		✓	✓			
Standard RAG			✓	✓										
OpenAI's Built-in RAG												✓		
OpenAI's Built-in Supervised Fine-tuning														✓
Well-developed Error-reporting System	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓

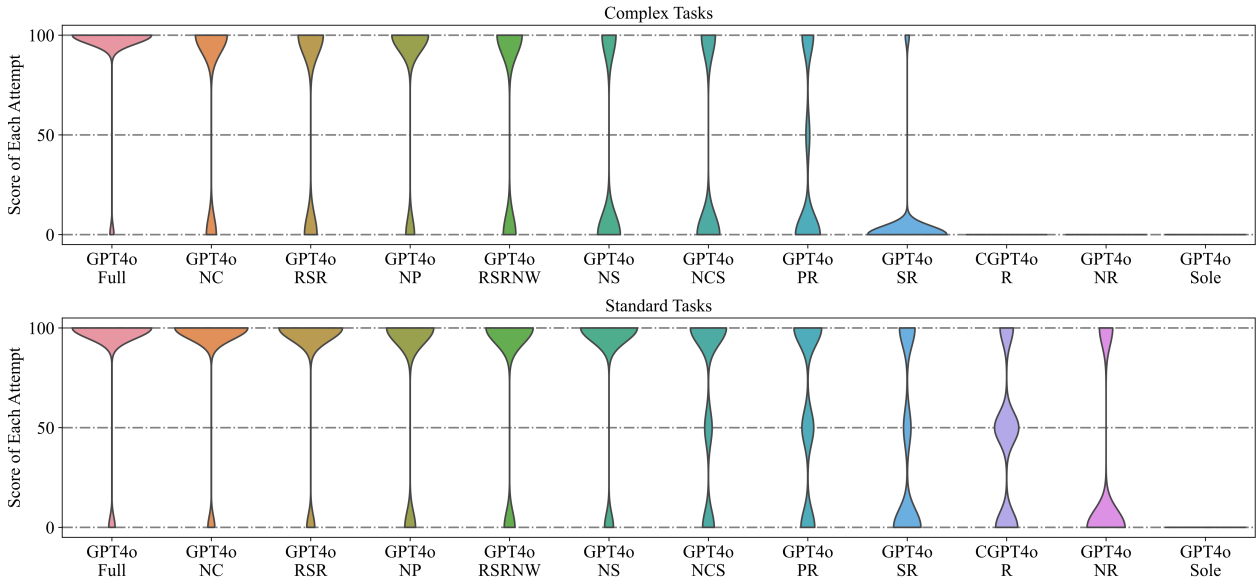

Fig. 7: Distribution of scores achieved across attempts for each evaluated scheme. For the definitions of scheme notations (e.g., `GPT4o-Full`), refer to Table I; the notations apply hereafter. (simulation environment: DALINE; GPT4o version: gpt-4o-2024-05-13; open-source repository for all tasks and results: [link]).

Comment 1.5: In Fig 8, what are the relationships between "All", "All-Complex", "All-Standard", "First Attempt", and "Final Attempt"?

Response to Comment 1.5*:* We agree that the previous caption of Fig. 8 may have been vague. To clarify, "All" refers to the aggregated success rate across all evaluated tasks, which includes both standard and complex tasks. "All-Complex" and "All-Standard" report the success rates calculated exclusively for complex and standard tasks, respectively. "First Attempt" represents the success rate achieved on the very first attempt for each task, before any error-feedback correction is applied, while "Final Attempt" reflects the success rate after all permitted attempts (i.e., up to the maximum number of feedback loops), capturing whether the task is eventually solved through iterative corrections. In summary, the first three categories report success rates by task type, and the latter two reflect success rates before and after error correction.

To make this relationship clearer, we have revised the caption of Fig. 8 accordingly in the manuscript and have also specified that this interpretation applies consistently throughout the entire manuscript:

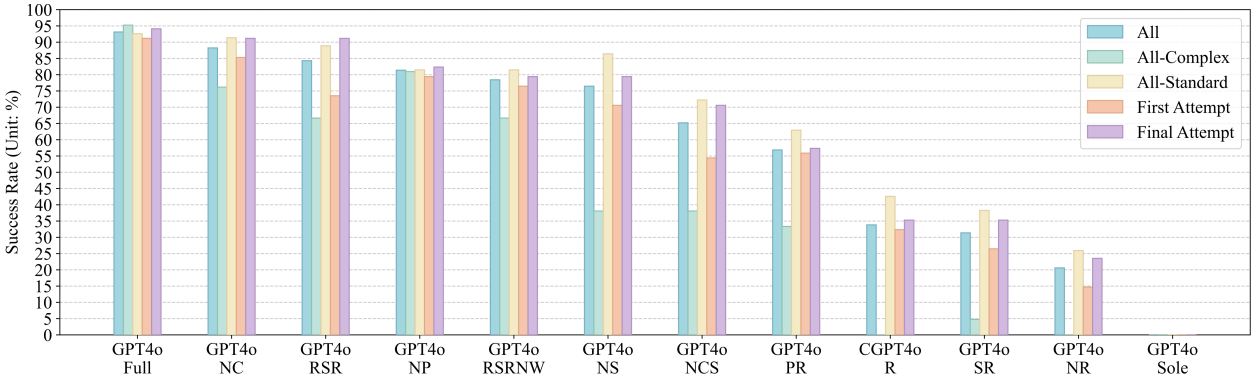

Fig. 8: Success rates for each scheme. "All" refers to the aggregated success rate across all tasks; "All-Complex" and "All-Standard" report the success rates calculated exclusively for complex and standard tasks, respectively. "First Attempt" represents the success rate achieved on the first attempt, before any error correction, and "Final Attempt" reflects the success rate after all permitted attempts are completed. The same interpretation applies hereafter (simulation environment: DALINE; GPT4o version: gpt-4o-2024-05-13; open-source repository for all tasks and results: [link]).

In addition, we also clarified this definition at the beginning of Section V.B to further improve consistency and readability, as shown below:

Section V.B. Evaluation on DALINE

...

Fig. 8 presents the success rates for each scheme, itemized by "all tasks combined", "complex tasks only", "standard tasks only", as well as for the "first attempt success rate" and the "final attempt success rate".

...

Comment 1.6: As a minor comment, I suggest adding indexes for the remarks in this paper. Also, there is a grammar error in Section IV-B. In the sentence, "This design not only supports automatic error correction but also improve adaptability to diverse simulation platforms," "improve" should be "improves".

Response to Comment 1.6: We thank the reviewer for the helpful suggestions. We have added indexes for the remarks throughout the manuscript as recommended. In addition, the grammar error noted in Section IV-D has been corrected. To further improve the quality of writing, we have carefully proofread the entire manuscript and applied grammar-checking tools to address similar issues. All revisions have been highlighted in red in the revised manuscript for clarity.

Response to Reviewer 2

General Comments to the Author: This paper is well-written, and the proposed method is novel and effective for intelligent power system simulation. In addition, the revised and added explanations concerning the adaptability and case studies improve the readability and reproducibility.

Response to the General Comment: We thank the reviewer for the positive and encouraging feedback. We also greatly appreciate the constructive suggestions provided during the previous review cycle, which have been invaluable in strengthening the overall quality of this work. We are very pleased that the reviewer recognizes both the novelty and effectiveness of the proposed method, as well as the improvements made to enhance the adaptability, readability, and reproducibility of the manuscript.