Response to Discussion 1: Domain Knowledge Integration and Production Readiness

Team LAMBDA
The Hong Kong Polytechnic University
Hong Kong SAR, China

We are grateful to Dr. Sammi Tang and Dr. Xuewei Wang for their thoughtful and comprehensive discussion of our work on LAMBDA [Sun et al., 2025a]. Their comments span several critical aspects of our system design, including domain knowledge integration, human intervention mechanisms, engineering considerations for production readiness, performance benchmarking strategies, and support for additional programming languages. We address each of these points in detail below.

As outlined in our survey on large language model-based agents for statistics and data science [Sun et al., 2025b], the field has seen significant advances in applying AI agents to data analysis tasks. LAMBDA represents one of the latest contributions to this rapidly evolving landscape.

1 Domain Knowledge Integration

Drs. Tang and Wang raise several important considerations regarding the Knowledge Integration Mechanism in LAMBDA. They correctly note that the performance of knowledge integration depends on multiple design factors: the contents of the key-value knowledgebase, the choice of embedding functions, similarity measures, and threshold settings. These observations align with our ongoing research agenda to strengthen domain knowledge integration.

Regarding embedding model selection, we agree that different transformer-based embedding models yield varying degrees of semantic similarity. In our subsequent work, we have explored contrastive fine-tuning approaches to improve retrieval accuracy. Specifically, we have investigated symmetric fine-tuning, which fine-tunes the encoder to process both queries and knowledge entries, and asymmetric alignment, which aligns the embedding model's representation space with that of a downstream task-oriented large language model. Our experiments demonstrate that fine-tuned models achieve substantial improvements over baseline models, with the asymmetric alignment approach achieving a 26.2% gain over the baseline on Hit Rate@1 metrics.

On the question of similarity thresholds, we acknowledge that relying on a fixed threshold ($\theta = 0.2$) is a heuristic that may not generalize across all domain-specific tasks. A fixed threshold risks either retrieving irrelevant code snippets (introducing noise) or missing critical context when the semantic overlap is subtle. To address this, future versions of LAMBDA

will move toward adaptive retrieval mechanisms. Instead of a static hard cutoff, we plan to implement a confidence-based retrieval system, where the agent evaluates the uncertainty of its internal knowledge before querying the knowledge base, effectively creating a dynamic threshold tailored to the complexity of the user's instruction.

We also recognize that cosine similarity is not universally optimal for assessing alignment quality, particularly when embedding magnitudes carry semantic information. Our empirical evaluations comparing multiple similarity measures (cosine similarity, Euclidean distance, Jaccard similarity, and overlap coefficient) reveal that vector-based metrics substantially outperform lexical matching methods. However, the performance gap between cosine similarity and Euclidean distance is marginal (less than 0.5% in MRR@10), indicating that the primary driver of performance is the quality of the embedding space rather than the specific distance function employed.

Concerning the distinction between "Full" and "Core" integration modes, we agree that the relationship between task complexity and integration level deserves closer examination. The Full mode provides more complete support but also leads to higher latency and greater token usage. We are considering a new architecture that includes a Routing Agent for the future version of LAMBDA. This agent would analyze the user's intent and decide how much knowledge integration is needed, choosing the Core mode for routine dataframe operations and switching to Full mode for more complex, domain-specific modeling tasks.

Additionally, we appreciate the suggestion to introduce quality controls for knowledge entries in the knowledgebase. User-supplied code descriptions vary significantly in clarity and quality. To involve quality control mechanisms, we plan to introduce an automated Quality Controls Agent that would utilize an LLM to standardize and rewrite user-contributed knowledge entries before they are indexed.

2 Human Intervention and Transparency

We are pleased that Drs. Tang and Wang recognize the value of LAMBDA's human-in-the-loop architecture. The dual-agent design (Programmer and Inspector) enables users to verify intermediate steps in the analytical pipeline. By presenting intermediate analysis codes and results at each step, the platform facilitates transparency and user control. This empowers users to debug and iterate effectively and builds trust in the system's outputs.

The automated report generation functionality, which captures prompt history, outputs, and visualizations into structured reports, addresses the critical need for documentation in regulatory, grant, and academic settings. We agree that integrating this feature with version control systems or collaborative platforms such as GitHub or JupyterHub would elevate its utility further and could transform LAMBDA into a more holistic data science platform.

Future iterations of LAMBDA could include more explainable AI features or guided walkthroughs for less experienced users. The integration of interactive visualizations, derivations, and literature links would enhance the educational value of the system.

3 Engineering Considerations for Production

The observation that LAMBDA's code execution success rate increased from 68.06% (with programmer alone) to 95.37% (with the dual-agent setting) demonstrates the effectiveness of our self-correcting mechanism. However, we acknowledge that further efforts are required to improve reliability for production readiness.

The remaining 4.63% of failed executions may stem from various factors, including limitations in model capability, ambiguous user instructions, or edge cases not adequately covered in the knowledge base. We are exploring additional corrective mechanisms, including more sophisticated error classification and specialized recovery agents that can handle specific categories of failures.

We also recognize that successful code execution does not always guarantee intended analytical outcomes, given known limitations and bugs associated with LLM-generated code. We plan to conduct additional reliability evaluations to assess the impact of such bugs through pressure testing in edge-case scenarios. Extending the Inspector's capabilities to verify analytical intent, such as appropriate use of variables or model assumptions, would enhance system reliability further. This would move the Inspector beyond syntactic correction toward semantic validation.

4 Performance Benchmarking with Domain-Aware Strategies

Drs. Tang and Wang correctly note that the three high-dimensional datasets tested in our experiments all come from transcriptomic studies, and LAMBDA consistently applied principal component analysis (PCA) for dimension reduction. The modest accuracy levels achieved may reflect limitations of PCA in such contexts. In fact, PCA is often suboptimal for transcriptomic data, and the machine learning community has developed numerous methods to address the small-n-large-p problem, with some specific to analyzing transcriptomic data.

This observation highlights a broader challenge: enabling domain-specific strategies in LAMBDA or similar systems. Various domains have developed specialized analysis strategies or methodologies to address their specific data or analysis challenges. To achieve this, it is crucial to expand evaluation with a broader range of domain-specific datasets or use cases. Such rigorous validation would require cross-disciplinary collaboration and could be facilitated through community-driven benchmarking or crowdsourced testing initiatives.

Motivated by this insight, we have begun developing a Data-Aware Retrieval framework that synthesizes user intent with statistical characteristics of the dataset. Rather than treating the dataset as a passive input, we make it an active component in the retrieval process. Specifically, retrieval is conditioned on a Data Profile—a vector that encodes key meta-features of the dataset, such as dimensionality, skewness, sparsity, and correlation structure.

5 Expansion with Data and Programming Support

We appreciate the suggestion to integrate data engineering capabilities into LAMBDA. Data preparation remains a major component of data science practice, and recent advances in generative AI are increasingly being applied to streamline data engineering tasks. The development or integration of data engineering agents would substantially extend LAMBDA's capability for end-to-end workflows from raw data to analytical insights.

Regarding multi-language support, we acknowledge that restricting LAMBDA to a Python-only environment limits its adoption within the broader statistical community, where R remains a dominant language. We are actively working on expanding the supported programming languages and software within the environment. In future versions, we will integrate a formal sandbox that will enable LAMBDA to execute shell commands as well as Python, R, Julia, SQL, and perform file-retrieval actions.

The hybrid programming workflow suggested by Drs. Tang and Wang—allowing data preprocessing in Python followed by statistical modeling in R, or vice versa—can be naturally supported by this sandbox framework. For example, an agent could perform data preprocessing in Python and conduct statistical modeling in R (using specialized packages such as `lme4`) within the same session. Such capabilities would significantly enhance the flexibility and applicability of LAMBDA.

We also note recent efforts to develop R-specific analysis agents, such as LLMAgentR and mergen, which provide promising avenues for extensions. Integration of these tools into the LAMBDA framework would further its mission of democratizing data analysis and supporting reproducible, domain-adaptable research workflows.

6 Conclusion

We thank Drs. Tang and Wang for their insightful comments and constructive suggestions. Their discussion has helped us identify several important directions for future development, including adaptive retrieval mechanisms, confidence-based knowledge integration, data-aware retrieval strategies, multi-language support, and enhanced quality control for knowledge bases. We are committed to addressing these challenges in our ongoing research agenda.

References

Maojun Sun, Ruijian Han, Binyan Jiang, Houduo Qi, Defeng Sun, Yancheng Yuan, and Jian Huang. Lambda: A large model based data agent. *Journal of the American Statistical Association*, pages 1–13, 2025a.

Maojun Sun, Ruijian Han, Binyan Jiang, Houduo Qi, Defeng Sun, Yancheng Yuan, and Jian Huang. A survey on large language model-based agents for statistics and data science. *The American Statistician*, pages 1–14, 2025b.