Paper ML20

Deterministic Foundations, AI Acceleration: Lessons from Developing Hybrid Clinical Review Tools

Matthew Kumar, Bayer Pharmaceuticals, Toronto, Canada
Srinivas Veeragoni, Bayer Pharmaceuticals, Whippany, USA

ABSTRACT

As large language models (LLMs) gain traction in clinical development, teams face a fundamental question: when should deterministic programming suffice, and when can probabilistic AI accelerate insight generation? We developed a medical review R Shiny application using a hybrid approach that combines robust, deterministic components for core analyses with AI-assisted exploration that accelerates medical reviewers' ability to address ad-hoc, novel questions with flexibility and velocity. This development journey revealed critical trade-offs—speed versus reproducibility, flexibility versus consistency, and breadth versus control—while highlighting the importance of transparency, human oversight and building (and maintaining) end user trust. We discuss practical consequences of the hybrid choice and propose suitability criteria based on problem definition, risk, criticality, and required traceability. Our experience demonstrates how clinical and development teams can harness both approaches while defining new ways of developing research tools that maintain the rigor required in regulated environments.

INTRODUCTION

In early-phase oncology clinical development, time is a critical resource. Oncology trials are complex due to novel compounds, cutting-edge trial designs, and critically ill patients. In this context, the ability to quickly understand trial data is not just an operational efficiency; it directly impacts pivotal GO/No-GO decisions on whether to continue the study. Rapidly generated insights can inform critical development choices and enable timely termination (or pivot) of a study if supporting evidence indicates it is unsafe or ineffective. Despite this urgency, the legacy tools or processes available for medical data review and exploration can contribute to delays. In some cases, the solutions in place may be inefficient for daily use or too rigid to adapt to the unique scenarios that arise in each study, particularly given oncology's novel trial design elements.

When these tools or processes are insufficient, clinical scientists and medical reviewers must resort to a more traditional process of requesting custom analyses. This standard process—from request specification to programming, validation, and delivery—can take up to several days. Such a delay represents a significant challenge to the rapid, iterative data exploration that time-sensitive decision-making requires. Often, existing tools are pre-specified and configured in a way that limits degrees of exploration (i.e., ‘hardcoded’ variable choices) functioning more as a presentation layer than a true exploration tool.

To address this critical need, an R Shiny application was developed with the initial goal of creating a deterministic, self-service tool to empower clinical scientists. By providing a reliable set of generic, yet flexible, interactive visualizations and tables, it increased the velocity and freedom of exploration for a core set of analyses for a given study.

While building this foundation, the parallel development of Large Language Models (LLMs) integrations in the R ecosystem introduced new possibilities to the project. It raised an interesting question: beyond the deterministic core, could the inclusion of LLMs accelerate the exploration of novel, ad-hoc questions that are difficult to pre-program, even with R and Shiny? Our focus was not on replacing the enhanced capabilities of the R Shiny application, which already offered greater flexibility and timeliness compared to prior solutions, but on carefully assessing the feasibility of how LLMs could complement our efforts in meeting the unique demands of early-phase oncology studies.

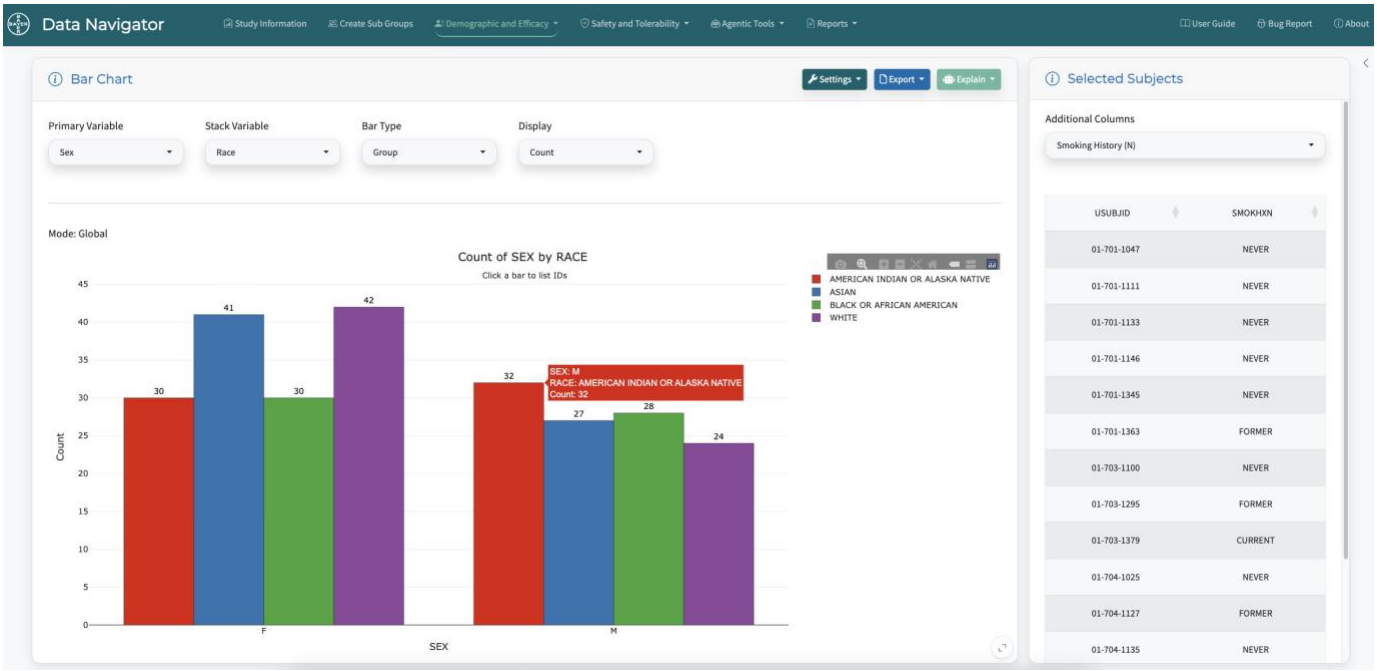
This paper shares our journey, highlighting practical lessons learned from building a hybrid tool, starting with the deterministic foundation and progressing to experiments integrating probabilistic approaches for data querying, interpretation, and visualization. It examines the trade-offs encountered and hopes to reach other development teams contemplating building similar applications.

BUILDING A DETERMINISTIC FOUNDATION WITH R/SHINY

We chose R and Shiny as the technology stack for our initial solution to support data review and exploration in early-phase oncology trials. The decision was based on their reactive framework and access to a well-established set of

data science packages, which make interactive visualizations and table displays possible. R and Shiny are also widely used in the pharma data science space, supported by community efforts like Pharmaverse[1]—a cross-industry open-source initiative creating R packages tailored for pharma—and successful pilots submitted to the FDA involving both R and Shiny[2]. Additionally, our internal Posit infrastructure streamlines the development, deployment, and sharing of these applications in a secure manner.

At its core, the application functionality revolves around a set of visualization and table pages, built jointly between clinical and data scientists to meet analysis needs. The starting focus was on exploration of baseline patient characteristics and efficacy data. Most content are built as modules that generate generic outputs, enabling scientists to choose variables and apply filters to match their analytical goals. This covers basic descriptive outputs like box plots, bar plots, and summary tables for identifying and stratifying patients or subgroups. It also includes more specialized visuals (e.g., efficacy) like swimmer plots, and shift tables with targeted features (e.g., % thresholding). Since these kinds of outputs aren't typically available in off-the-shelf solutions, they needed custom and careful programming to meet specifications and requirements and to ensure they scale across studies.



To improve the experience for reviewers in early-phase oncology trials, the application includes several convenience features. A two-tier filtering system (with global and local scopes) allows for detailed segmentation across all visuals and tables, helping dive deeper into study data for focused exploration. Export options to static outputs (e.g., pptx, docx), including both single items and complete decks, assist with internal discussions by including metadata (e.g., active filters applied and subject identifiers) to maintain traceability. We also employed the *shinymeta*[3] package in the application, which lets users see and save the underlying R code for visuals and tables they create in the tool. This provides a record of the interactive analysis and makes it easier to verify, adjust, or rerun analyses in different R environments.

The design of the application prioritizes flexibility and scalability to handle the range of challenges and differences seen in early-phase oncology trials. Its adaptable structure, built with reusable Shiny modules, allows it to adjust to various scenarios. For example, the stacked bar graph module can be used to visualize demographic characteristics, but also to display adverse events by toxicity grade just by making selections in the UI. Configuration through YAML files was first configured for the primary data model used in medical review, and setup occurs on a per-study basis. This approach has shown it can easily extend to other models, like those used in late-stage clinical development (e.g., ADaM), opening doors to new use cases or data sources (e.g., biomarker).

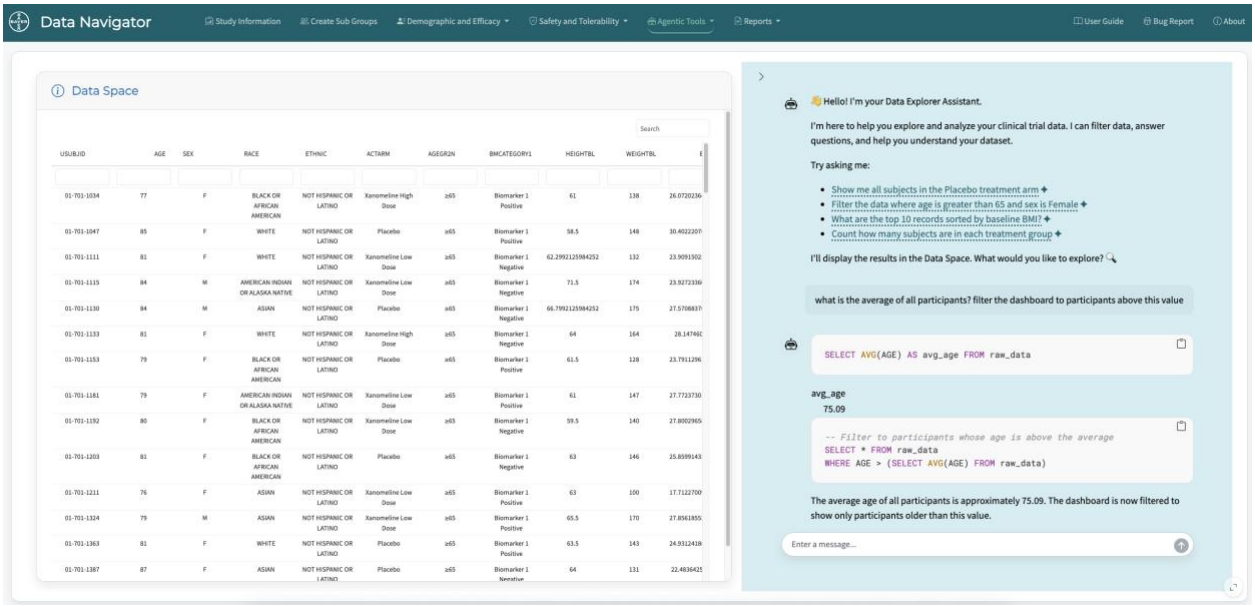
The development process includes the user base at the center: starting with early specification and feedback from clinical scientists and refining the application through iterative design. Prototypes were shared frequently to ensure the application matched user workflows, helping with adoption and meeting real needs in medical review and exploration for early-phase oncology.

During pilot testing on real studies, we noticed the (anticipated) variation in trial designs and requirements and its impact on the data and application-level elements. We jointly recognized a shared need for study-specific adjustments to manage unique designs, variations, or personal review preferences. Although the application's modular design covered core needs well, meeting every individual request proved challenging due to inherent variability. This prompted us to think about approaches where we could balance addressing primary analysis needs while still handling specific requirements or preferences. At this point, we turned to LLMs as a possible mechanism to assist us.

LLM COMPONENTS: MOTIVATION AND INITIAL EXPLORATION

Our exploration into LLMs began with a simple observation during pilot testing of users' interactions with the application. Using the two-tier filter system, users could easily identify a subset of participants for simple criteria (e.g., filter between ages 18 and 40 years). However, for more nuanced criteria (e.g., filter participants above the mean age of their treatment arm), it was obvious our application could not accommodate this request without additional R and Shiny programming. While a dedicated filter could have been developed, it was evident that users might require even more tailored options based on study-specific criteria. Programming each possible variation was impractical and laborious, highlighting the need to investigate potential solutions to bridge the gap between static programming and dynamic, ad-hoc user requirements.

This observation coincided with recent advancements in the R ecosystem, where integrations with LLMs were becoming more accessible through packages such as *ellmer*[4] and *shinychat*[5]. Our initial exploration into LLMs involved implementing a chat feature using an early version of the *querychat*[6] package. This implementation allowed users to interact with study data by engaging in a conversation using natural language. For a typical user experience, a request like "filter observations above the mean age of the sample" can be typed directly into the chat interface. The system then executes the corresponding query, dynamically updating the data table in the application for the user to view the results and continue their exploration. As a side note, if additional plots or other outputs are wired to this reactive dataset, they too would update automatically. Another capability enabled by the querychat package was performing custom data analysis. For example, calculating a geometric mean (or some other measure) summary for a continuous variable, which wasn't explicitly programmed but could be of interest in certain analyses. In other instances, obtaining row percentages instead of column percentages based on user preference, despite an existing table creation module. It's clear that this approach could potentially offer significant benefits in speeding up tailored data exploration, but at what trade-offs?



LLMs are probabilistic in nature, meaning there is no guarantee of correctness or reproducibility in their outputs. To build trust in this technology, it's important to understand its limitations and use it with a clear awareness of what it can and cannot do effectively. *Querychat* includes a feature to display the underlying SQL query generated from natural language input and applied to the data clearly on the chat interface. Many users in this domain possess sufficient data familiarity to assess the accuracy of these queries or, when necessary, consult a statistical programmer or data scientist for validation. Even with this additional review, the process is considerably faster than traditional, manual request workflows. This instills an "ask, review, trust but confirm" approach, encouraging users to use the tool for quick insights while verifying results offline for critical decisions.

Overall, user feedback has been initially positive, highlighting that the feature was intuitive and useful. However, instances of misinterpretation due to ambiguously worded user queries highlighted a clear need for guidance on formulating precise questions. This involves potentially refining system prompts, providing additional instruction sets or tweaking LLM settings to better interpret user intent. To this end, we've also included "suggestions" in the chat interface when the application initially loads. This can help users not only understand what is possible but also how to frame their requests effectively.

Limitations were also encountered with more complex data operations that relate to filtering and data analysis. One such example is the necessity of joins across domains (e.g., integrating demographics with laboratory data tables) to answer complex questions. The differing table structures (e.g. single row vs multi row per subject) presented obstacles that we could not consistently overcome. At this stage, the features proved more effective for simpler, single data frame tasks. Therefore, we suggest pre-processing data beforehand as a good alternative strategy to reduce complexity. For example, simplifying datasets by flattening or pre-aggregating variables has demonstrated more utility in the LLM feature.

Due to the sensitive nature of clinical study data, ensuring security was a top priority, even in our ideation phase. Our tool is linked to a secure internal service for LLM interactions. Furthermore, when users engage with study data, only the data schema is shared with the model, not the data itself. Function and tool calling enable the LLM-generated SQL query to be executed on our systems, preventing offsite processing and preserving control over sensitive information. Additionally, the LLM feature was developed as an experimental module, allowing it to be easily enabled or disabled without affecting the core application source code.

As our tool evolved, we implemented a feedback mechanism to track the most frequently requested features, including "interesting finds" through user interactions with LLMs. We conduct regular meetings with end users to gather insights on what they found beneficial or necessary, evaluating whether to integrate these elements deterministically into the application. For instance, the option to filter by custom percentiles emerged as a key need. The LLM functionality was instrumental in revealing "what's feasible" and pinpointing user specifications for future improvements.

EXPANDING LLM USE: INTERPRETATION AND CUSTOM TOOLS

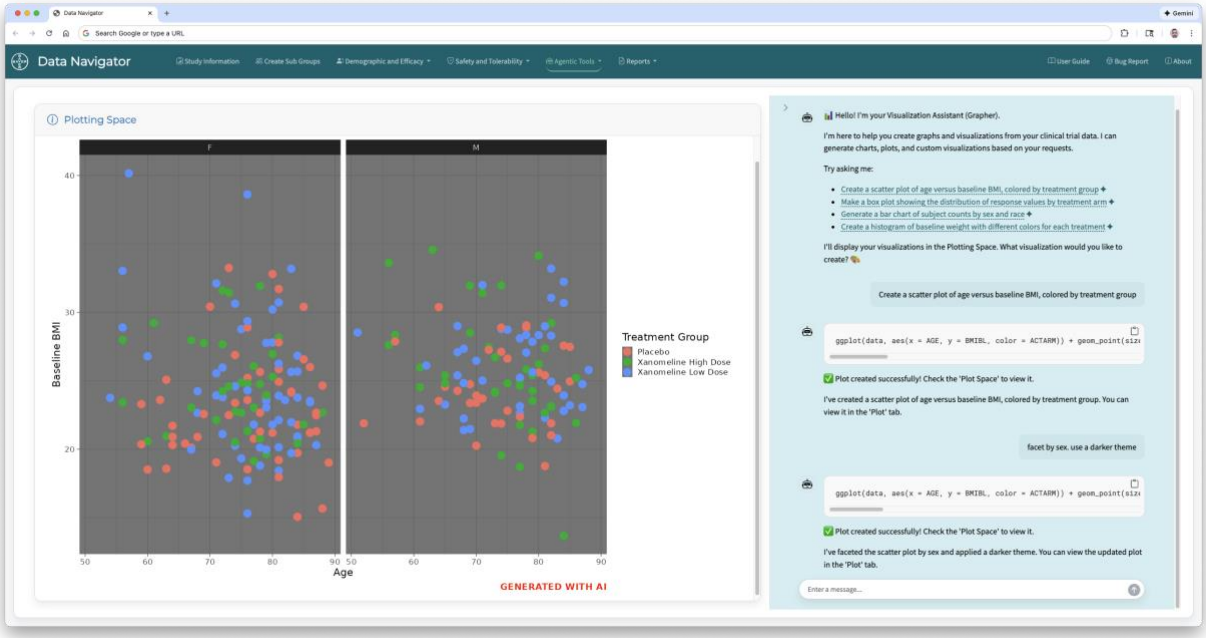
Building on the initial exploration and positive reception of the LLM chat feature, we began exploring additional functionalities to enhance the application to accelerate the delivery of timely insights. Our efforts focused on performing LLM-assisted inference from visualizations and table results to gain deeper insights.

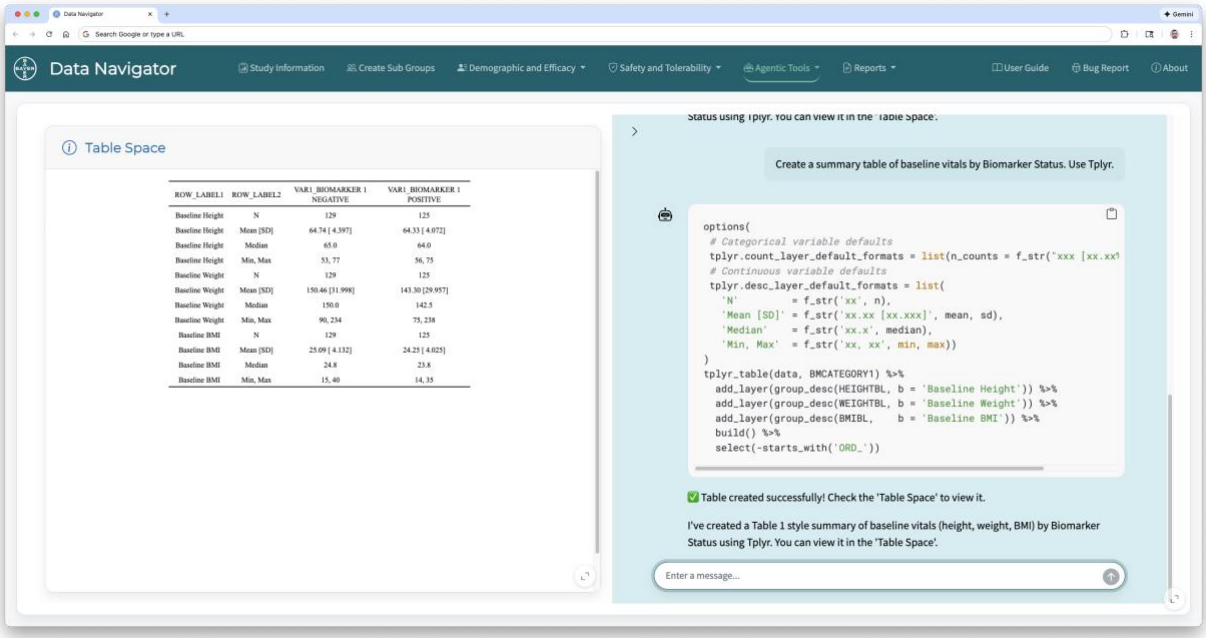
As a next step, we piloted a feature[7] that allows users to receive an LLM-assisted interpretation of their created content, such as bar graphs or summary tables, with a single click of a button in the application. We provided additional context by including user settings, such as applied filters. For example, it can spot trends like imbalances in bar graphs or differences in line graphs, acting as a secondary pair of eyes. This helps reviewers get quick confirmation or a new perspective without a lengthy secondary review. However, the feature's usefulness is limited because it operates somewhat in a "vacuum", focusing on data without grasping the full study design context and analysis complexities.

Parallel to interpretation, we simultaneously explored LLM-assisted plot and table creation to empower users with even greater flexibility. Using the function and tool-calling approach, we developed extensions that allowed users to create novel plots (using *ggplot2*[8]) and tables (using *Tplyr*[9]) directly within the application. We adapted an element of transparency by providing the generated R code on the user interface, mirroring the approach used with SQL queries in the filter feature. The user benefit was obvious: it enabled the creation of unsupported visuals, such as a scatter plot not available in our deterministic modules, or extending existing ones, such as a stacked bar (e.g., race by sex) graph faceted by a third variable (e.g., biomarker status). For tables, it supported custom statistical summaries, which, when paired with our export to PowerPoint flexibility, accelerated data presentation for diverse reviewer needs.

Despite these advancements, limitations persisted with niche outputs critical to oncology studies. Specifically, the system struggled with specialized clinical plot requests like waterfall plots, spider plots, or Kaplan-Meier curves, which require nuanced methodology (e.g., statistical specifications) and complex data prerequisites (e.g., wrangling, merging, transformations). Similarly, intricate tables (e.g. shift), which often involve multi-dataset merges, were not reliably handled by this implementation at this stage. Due to the criticality of these outputs (i.e., efficacy), deterministic programming remains the preferred approach to ensure accuracy and reliability for high-stakes results. Another barrier was user education: technical jargon, such as “faceting” in *ggplot2*, was unfamiliar to many users, limiting the effectiveness of their requests for examining relationships by a third factor. Users also struggled with how to articulate desires for changes in other plot aesthetics or features, or how to request specific table layouts like spanning headers. Again, we partially addressed this by incorporating system prompts and preloaded suggestions in the chat UI to demonstrate query phrasing, but ongoing support is needed for broader utility. Finally, as a safeguard, LLM-generated graphs and tables carry watermarks to prevent circulation as definitive findings, reinforcing caution and mitigating the risk of unverified outputs influencing clinical decisions.

These dual explorations into LLM-assisted interpretation and custom tool generation represented a meaningful advance in our journey, balancing innovative potential with the practical constraints of a regulated environment. They established a foundation for further evaluating how LLMs could complement deterministic methods while addressing the inherent trade-offs.


Data Navigator
Study Information
Create Sub Groups
Demographic and Efficacy
Safety and Tolerability
Agentic Tools
Reports
User Guide
Bug Report
About
Plotting Space
F
M
Baseline BMI
Age
Treatment Group
Placebo
Xanomeline High Dose
Xanomeline Low Dose
GENERATED WITH AI
Hello! I'm your Visualization Assistant (Grapher).
I'm here to help you create graphs and visualizations from your clinical trial data. I can generate charts, plots, and custom visualizations based on your requests.
Try asking me:
Create a scatter plot of age versus baseline BMI, colored by treatment group
Make a box plot showing the distribution of response values by treatment arm
Generate a bar chart of subject counts by sex and race
Create a histogram of baseline weight with different colors for each treatment
I'll display your visualizations in the Plotting Space. What visualization would you like to create?
Create a scatter plot of age versus baseline BMI, colored by treatment group
Plot created successfully! Check the 'Plot Space' to view it.
I've created a scatter plot of age versus baseline BMI, colored by treatment group. You can view it in the 'Plot' tab.
facet by sex. use a darker theme
Plot created successfully! Check the 'Plot Space' to view it.
I've faceted the scatter plot by sex and applied a darker theme. You can view the updated plot in the 'Plot' tab.
Enter a message...


Data Navigator
Study Information
Create Sub Groups
Demographic and Efficacy
Safety and Tolerability
Agentic Tools
Reports
User Guide
Bug Report
About
Table Space
ROW_LABEL1 ROW_LABEL2 VAR1_BIOMARKER 1 NEGATIVE VAR1_BIOMARKER 1 POSITIVE
Baseline Height N 129 125
Baseline Height Mean [SD] 64.74 [4.397] 64.33 [4.072]
Baseline Height Median 65.0 64.0
Baseline Height Min, Max 53, 77 56, 75
Baseline Weight N 129 125
Baseline Weight Mean [SD] 150.46 [31.998] 143.30 [29.957]
Baseline Weight Median 150.0 142.5
Baseline Weight Min, Max 90, 234 75, 238
Baseline BMI N 129 125
Baseline BMI Mean [SD] 25.09 [4.132] 24.25 [4.025]
Baseline BMI Median 24.8 23.8
Baseline BMI Min, Max 15, 40 14, 35
Status using Tplyr. You can view it in the 'Table Space'.
Create a summary table of baseline vitals by Biomarker Status. Use Tplyr.
options(
# Categorical variable defaults
tplyr.count_layer_default_formats = list(n_counts = f_str("xxx [xx.xx
# Continuous variable defaults
tplyr.desc_layer_default_formats = list(
'N' = f_str('xx', n),
'Mean [SD]' = f_str('xx.xx [xx.xxx]', mean, sd),
'Median' = f_str('xx.x', median),
'Min, Max' = f_str('xx, xx', min, max))
)
tplyr_table(data, BMCATEGORY1) %>%
add_layer(group_desc(HEIGHTBL, b = 'Baseline Height')) %>%
add_layer(group_desc(WEIGHTBL, b = 'Baseline Weight')) %>%
add_layer(group_desc(BMIBL, b = 'Baseline BMI')) %>%
build() %>%
select(-starts_with('ORD_'))
Table created successfully! Check the 'Table Space' to view it.
I've created a Table 1 style summary of baseline vitals (height, weight, BMI) by Biomarker Status using Tplyr. You can view it in the 'Table Space'.
Enter a message...

BALANCING DETERMINISTIC AND LLM APPROACHES: KEY LESSONS LEARNED

Developing our hybrid clinical medical review and exploration application yielded valuable insights, particularly in early-phase oncology. We identified critical trade-offs that apply across various contexts, helping to determine when deterministic programming or probabilistic LLM methods are most effective for specific tasks. These lessons guide the purposeful integration of hybrid approaches in clinical settings.

One critical trade-off is Speed vs. Reproducibility. LLM-driven features, like natural language query filtering, enable rapid data exploration and custom insights, significantly reducing development time for ad-hoc analyses. This speed is vital in fast-paced oncology trials, especially for straightforward or less complex tasks. However, these probabilistic LLM outputs require verification. In contrast, deterministic tools, such as pre-programmed swimmer plots, ensure consistent, predictable, and verifiable results, which are critical for standardized analyses requiring exact reproducibility, formal validation, and audit trails—particularly in high-stakes areas like efficacy reporting.

Building on this, another key consideration is Flexibility vs. Consistency. LLMs excel at handling dynamic, user-driven queries, offering adaptability for unique study questions, such as generating novel content on demand. Yet, this flexibility can easily lead to inconsistency. For instance, an LLM might misinterpret a request for a specialized clinical plot, producing a candlestick plot instead of a waterfall plot, resulting in inaccurate outputs. Deterministic Shiny modules on the other hand, provide a reliable foundation by consistently meeting established analytical needs. A hybrid strategy is essential: LLMs can enhance exploratory capabilities and shorten development cycles for general tasks, while critical outputs can be supported by potentially integrating mature deterministic functions (e.g., waterfall plot logic) as LLM-accessible tools via natural language requests. Rigorous testing and system prompt engineering are also necessary to optimize LLM models for clinical language and bridge user jargon gaps. One promising approach to help support this effort is the *vitals*[10] R package, offering a principled framework to structure, execute, and review LLM evaluations.

Finally, a critical trade-off we considered was Breadth vs. Control. LLMs offer exploration and interpretation, uncovering new insights in areas difficult to pre-program. However, they often lack the precise control of deterministic tools, which is crucial in settings requiring accuracy and accountability. For example, while LLM-assisted interpretation aids “sanity checks”, it operates in isolation, missing full study design context. Human-in-the-loop (HIL) oversight remains vital to provide nuanced clinical context to interpretation, especially for decision-critical outputs. Integrating study context through Retrieval-Augmented Generation (RAG) systems, such as the *ragnar*[11] package, is a promising step to incorporate content from key documents like protocols or data review plans, enhancing the relevance and control of LLMs. This underscores the need for a hybrid approach: leveraging LLMs for discovery while relying on deterministic methods for high-stakes decisions. Traceability further emphasizes this balance—deterministic tools like *shinymeta* work in towards audit trails, whereas LLM outputs require manual validation despite code transparency, influencing decision-making in clinical contexts.

Reflecting on these trade-offs, our experiences suggest that neither deterministic nor LLM approaches alone can fully meet the needs of clinical data review particularly when time is a factor. By thoughtfully combining them, we strive to create a more balanced approach that merges innovation with precision, offering a practical way forward for future development.

CONCLUSION

Looking ahead, our vision for early clinical development oncology positions this hybrid Shiny application, enriched with LLM-assisted features, as a cornerstone of an end-to-end disease data strategy. Recognizing that LLMs will touch many aspects of our operations, we aim to leverage them to foster collaboration and shared benefits. By uniting functions like data management, monitoring, insights generation, statistical programming, statistics, and clinical and translational study leads on a single platform, we begin with key study questions and front-load components from protocol and EDC building to submission reports. This comprehensive integration enables accelerated, effective, and efficient clinical trial conduct, transforming how data drives decision-making across the trial lifecycle.

Our journey with hybrid clinical review tools has revealed a powerful synergy between deterministic foundations and LLM-driven capabilities, each complementing the other to address the complex demands of early-phase trials. Deterministic methods provide the rigor and reliability essential in regulated environments, while LLMs accelerate custom insights, empowering users to explore data in innovative ways. Yet, context remains king; end users possess vital knowledge about study design and clinical nuances that LLMs cannot fully capture, highlighting the need for caution and oversight. This reinforces the importance of HIL processes in clinical tool development and its applications, ensuring technology enhances rather than replaces expert judgment. One additional avenue we are exploring is evaluating agentic approaches to take the probabilistic-generated code and output and run through a series of evaluations and converting it into a deterministic code. This is a work in progress. As we advance, we are dedicated to refining these tools to balance innovation with precision in clinical research.

REFERENCES

Lauderdale, K. (2025). *OS09: May the Code Be with You: Entering the pharmaverse Galaxy*. Presented at PhUSE EU Connect 2025.

R Consortium. (n.d.). *Using R to Submit Research to the FDA: Pilot 4 Successfully Submitted*. Retrieved January 5, 2026, from https://r-consortium.org/posts/using-r-to-submit-research-to-the-fda-pilot-4-successfully-submitted/

Cheng J, Sievert C (2025). *shinymeta: Export Domain Logic from Shiny using Meta-Programming*. R package version 0.2.1.9000, https://github.com/rstudio/shinymeta, https://rstudio.github.io/shinymeta/.

Wickham H, Cheng J, Jacobs A, Aden-Buie G, Schloerke B (2025). ellmer: Chat with Large Language Models. R package version 0.4.0, https://ellmer.tidyverse.org.

Cheng J, Sievert C, Aden-Buie G, Schloerke B (2025). shinychat: Chat UI Component for 'shiny'. R package version 0.3.0, https://posit-dev.github.io/shinychat/r/.

Aden-Buie G, Cheng J, Sievert C (2026). querychat: Filter and Query Data Frames in 'shiny' Using an LLM Chat Interface. R package version 0.2.0, https://posit-dev.github.io/querychat/r/.

Cheng, J. (n.d.). *r-sidebot*. GitHub Repository. Retrieved from https://github.com/jcheng5/r-sidebot/tree/main/R

H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

Miller E, Stackhouse M, Tarasiewicz A (2024). Tplyr: A Traceability Focused Grammar of Clinical Data Summary. R package version 1.2.1, https://github.com/atorus-research/Tplyr.

Couch S (2025). vitals: Large Language Model Evaluation. R package version 0.2.0, https://github.com/tidyverse/vitals..

Kalinowski T, Falbel D (2026). ragnar: Retrieval-Augmented Generation (RAG) Workflows. R package version 0.3.0, https://ragnar.tidyverse.org/.

ACKNOWLEDGMENTS

We would like to thanks our Data Science & AI, Early clinical development and Clinical development operations colleagues for supporting us in developing this R Shiny application.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Matthew Kumar
Bayer Inc
Data Science & Artificial Intelligence
2920 Matheson Blvd E
Mississauga, ON L4W 5J4
Canada
Matthew.Kumar@bayer.com

Srinivas Veeragoni
Bayer US LLC
Data Science & Artificial Intelligence
100 Bayer Boulevard, Whippany, NJ 07981-1544, United States
+1 (973) 610-0884
srinivas.veeragoni@bayer.com

Brand and product names are trademarks of their respective companies.