Paper ML14

From Automation to Audit-Readiness—AI's Growing Role in Statistical Programming

Manoj Kumar Maripally, Cytel, USA

ABSTRACT

AI-assisted statistical programming is transitioning from a theoretical concept to an operational reality, fundamentally reshaping the delivery of clinical development. By automating repetitive activities such as data cleaning, coding, and validation, AI technologies reduce operational burden and accelerate critical milestones, including SDTM and ADaM dataset development, clinical reporting, database lock, and regulatory submissions. Generative AI, particularly large language models, offers additional value in the early stages of code development and documentation drafting. Nevertheless, robust human oversight remains essential. AI should be positioned as a collaborative tool rather than a fully autonomous solution—enhancing programmer efficiency while preserving the expert judgment required for contextual interpretation, reproducibility, and regulatory compliance. This paper presents practical applications of AI-assisted programming in clinical development and outlines key guardrails, including validation frameworks, audit preparedness practices, and compliance strategies, to ensure responsible adoption aligned with regulatory expectations.

INTRODUCTION

Modern clinical development continues to rely heavily on manual statistical programming at a time when clinical trials are experiencing rapid growth in data volume, increasing protocol complexity, and heightened regulatory expectations. As a result, statistical programmers spend a disproportionate amount of time on data preparation, cleaning, and validation activities. These manual approaches introduce data-processing bottlenecks, require significant human resources for quality assurance and validation, and increase the risk of errors and inconsistencies in complex analyses. Moreover, the substantial documentation and compliance burden further strains programming capacity. Collectively, these challenges limit efficiency and pose quality risks, making it increasingly difficult for traditional workflows to meet compressed trial timelines, particularly the expectation to deliver outputs within 10–15 days following database lock.

AUTOMATION IN CLINICAL TRIALS STATISTICAL PROGRAMMING

Statistical programming for clinical trials remains highly labor-intensive and susceptible to error, particularly in the generation and validation of study-specific ADaMs and TLFs (tables, listings, and figures). In response, recent advancements have focused on automation-enabling foundations, including metadata-driven architectures that decouple specifications from implementation and open-source R ecosystems such as *pharmaverse*, which provide modular, reusable components aligned with CDISC standards, including SDTM IG 3.4, ADaM IG 1.3, Define-XML 2.1, and the Analysis Results Standard 1.0. Additional innovations include machine-learning techniques for automated data mapping, emerging applications of large language models for code generation, natural language processing to translate textual inputs into programmable specifications, and deep-learning methods for pattern detection and predictive insights. Validation processes are also evolving through risk-based automation strategies and CI/CD pipelines, alongside careful platform considerations for R, SAS, and Python within regulatory environments. Despite these advances, comprehensive automation of statistical programming—particularly for TLF production, validation frameworks, and AI-assisted code development—remains in an early and rapidly evolving stage.

THE PHARMAVERSE OPEN-SOURCE ECOSYSTEM

A comprehensive, end-to-end, production-ready tool chain is now available to support fully automated, CDISC-compliant clinical reporting. For dataset generation, R-based packages such as *admiral* provide extensive CDISC-aligned ADaM derivations through a robust library of reusable functions, while tools like *REDCap2SDTM* enable automated SDTM mapping with substantial efficiency gains. TLF generation is supported by metadata-driven frameworks including *rtables* and *Tplyr,* which deliver high-quality, reproducible outputs, alongside solutions such as TLFQC that facilitate codeless table production with integrated quality control. Submission and validation activities are addressed through tools such as *xportr, Pinnacle 21*, and *defineR*, enabling the creation of submission-ready XPTs and automated Define-XML 2.1 documentation. Workflow orchestration platforms, including *targets* and *Snakemake,* further enhance reproducibility through dependency-tracked, scalable pipelines. Collectively, these production-mature open-source tools demonstrate that modern clinical reporting can be fully automated, delivering faster timelines, improved quality, and scalable, metadata-driven workflows suitable for regulatory submissions.

METADATA DRIVEN TFL OUTPUT GENERATION

Clinical study reporting typically requires the production of hundreds of tables, listings, and figures (TFLs), which are subject to frequent revisions throughout study execution. Traditional manual programming approaches are therefore time-consuming, error-prone, and difficult to scale. To address these challenges, a SAS-based, metadata-driven framework known as *atlas* leverages CDISC Analysis Results Standard (ARS) metadata embedded directly within TFL shells to automatically generate fully executable TFL programs. In this approach, ARS metadata is defined prospectively within the shells and used to drive macro selection and parameterization, enabling *atlas* to produce ready-to-run SAS programs for each TFL. This framework delivers significant reductions in manual programming effort, ensures consistent and traceable submission-ready outputs, and allows rapid adaptation to TFL changes during study conduct. By shifting TFL development from a code-centric to a metadata-centric paradigm, *atlas* enhances programmer productivity while improving transparency and oversight across clinical reporting workflows.

VALIDATION FRAMEWORK

Traditional validation approaches in statistical programming have relied heavily on independent double programming to ensure result quality; however, this method is resource-intensive, highly redundant, and often results in late detection of errors that require costly rework. In contrast, risk-based validation frameworks apply ICH Q9 quality risk management principles to align validation effort with the criticality of outputs. Under this approach, high-risk analyses—such as primary efficacy and key safety endpoints—continue to receive full independent programming, while medium-risk outputs are validated through structured peer review supplemented by automated testing, and low-risk outputs rely primarily on automation. Risk assessments consider multiple dimensions, including output criticality, frequency of code changes, and the level of regulatory scrutiny, with analyses supporting labeling claims receiving enhanced validation. Additionally, independent raw-to-TFL programming by statisticians provides targeted oversight for critical outputs. Collectively, this framework optimizes quality assurance by focusing resources on high-impact areas while improving efficiency, transparency, and regulatory alignment.

AUTOMATION AND AI/ML APPLICATIONS

Automation in clinical development must be implemented in a manner that preserves ALCOA+ data integrity principles, ensuring that data and outputs are attributable, legible, contemporaneous, original, and accurate, while also remaining complete, consistent, enduring, and readily available. When appropriately designed, automated systems can strengthen ALCOA+ compliance through mechanisms such as immutable audit trails enabled by version control, automated provenance tracking, reproducible execution environments, and systematic generation of documentation. Current AI and machine learning applications in statistical programming include automated CDISC data mapping, AI-assisted code generation using large language models, protocol parsing through natural language processing to extract structured study information, and anomaly detection models to identify data quality issues. In the area of data mapping, effective machine learning approaches leverage a combination of variable-level attributes, contextual metadata, and semantic features derived from text embeddings, such as those generated by BERT. Empirical evaluations have shown that neural network models achieve the highest accuracy for complex mapping relationships, while random forest models offer greater interpretability through feature importance, supporting transparent and risk-informed adoption in regulated environments.

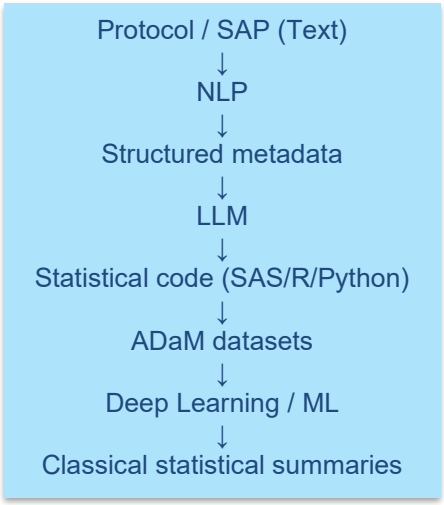
LARGE LANGUAGE MODELS FOR STATISTICAL PROGRAMMING

Large language models (LLMs), including general-purpose systems such as GPT-4 and Claude as well as domain-specific models like ClinicalBERT and GatorTron, are increasingly being applied to support statistical programming activities, including code drafting, interpretation of analysis specifications, and code review. While general-purpose models demonstrate strong capabilities in complex reasoning, domain-adapted models consistently outperform them on clinical natural language processing tasks. In statistical programming contexts, LLMs show higher performance for routine, standardized, and repetitive tasks, although real-world deployments have reported variable levels of success. Key limitations include reproducibility challenges arising from non-deterministic outputs, limited embedded knowledge of CDISC standards and therapeutic-area specifics without targeted fine-tuning, and the absence of formal regulatory guidance governing AI-generated code. Consequently, best-practice adoption positions LLMs as “first-draft assistants” rather than autonomous programmers, with their use aligned to risk-based validation frameworks and established software assurance principles to ensure quality, transparency, and regulatory compliance.

AI/ML HYBRID WORKFLOW

Large language models operate at the interface between human intent and statistical code, translating analytical requirements into executable programming constructs and thereby reducing manual development effort. Despite these

efficiencies, human validation remains essential to ensure correctness, interpretability, and regulatory compliance. Natural language processing plays a complementary upstream role by converting unstructured text—such as protocols and analysis plans—into structured inputs that can be consumed by statistical programming workflows. Deep learning, by contrast, is applied less frequently to classical statistical inference and more often to pattern detection and predictive tasks. Together, these approaches position AI technologies as enablers that augment statistical programming and analysis, complementing traditional statistical methods rather than replacing them.



AI/ML IMPLEMENTATION IN STATISTICAL PROGRAMMING

Example 1: TFL Automation	• NLP reads mock shell • LLM generates SAS code • Programmer reviews & validates • Output traced to ADaM
Example 2: QC Automation	• Deep learning flags unusual distributions • LLM explains why they're unusual • Programmer decides if it's real or an error

Example 3: Model Exploration	• Deep learning finds nonlinear effects
	• Statistician formalizes with:
	  • Cox model
	  • Mixed model
	  • Bayesian model

STRENGTHS VS LIMITATIONS OF AI/ML IMPLEMANTATION

In regulated environments, the application of AI must align with regulatory expectations and established Good Programming Practice (GPP), ensuring that AI-generated outputs are never treated as final without appropriate oversight. Full traceability must be maintained, with mandatory human review and sufficient model explainability or justification to support inspection and audit readiness. Within this framework, large language models are best positioned as programmer productivity tools rather than analytical decision engines, functioning as statistical programming copilots. Natural language processing serves as a bridge between unstructured text and structured data, while deep learning operates as a pattern discovery engine that complements traditional statistical methods. Together, these technologies accelerate programming activities, improve consistency, and expand analytical insight, while preserving statisticians' accountability and control over analyses and regulatory submissions.

Technique	Strength	Limitation
LLMs	Code & explanation speed	Hallucination risk
NLP	Extracts meaning from text	Domain tuning required
Deep learning	Complex pattern detection	Low interpretability
Classical stats	Interpretability, compliance	Less flexible

SUMMARY

Automation in statistical programming has reached a high level of technical maturity, driven by robust open-source ecosystems such as *pharmaverse*, active industry consortia including PHUSE, CDISC, and the R Validation Hub, and growing regulatory acceptance of R for clinical submissions. Despite this progress, adoption remains uneven: large pharmaceutical organizations have led implementation, while smaller sponsors often face resource and skill constraints. A notable disconnect persists in practice, where validation considerations continue to dominate automation efforts and influence adoption pace. Key enablers of successful implementation include strong metadata governance and standardization, reusable component libraries, modern software engineering practices such as Git-based version control, CI/CD pipelines, and automated testing, as well as phased rollouts supported by pilot studies and proven regulatory precedents, including FDA-reviewed R submissions. Conversely, major barriers include entrenched legacy SAS investments, migration inertia, skills gaps in transitioning to R, Python, and metadata-centric workflows, and regulatory uncertainty that fosters risk aversion. As automation advances, the role of the statistical programmer is evolving from manual code development toward metadata management, automation design, and quality engineering, with inspection readiness anchored in clear validation rationales, risk-based assessments, demonstrable reproducibility, end-to-end traceability from SAP to code, data, and outputs, and automation that strengthens ALCOA+ data integrity compliance.

While automation and AI continue to advance, their application in statistical programming remains constrained by limitations in domain-specific reasoning, contextual interpretation of results, and the need for rigorous human oversight. AI is therefore unlikely to replace statistical programmers; instead, it will automate routine and repetitive coding tasks while elevating human roles toward higher-order analytical thinking, interpretation, and decision-making. The longevity of the profession is secure, but its nature is undergoing significant transformation, with professionals who embrace change positioned for stronger career prospects and more impactful, value-driven roles. To remain competitive, upskilling priorities must extend beyond traditional programming to include proficiency in R and Python, data visualization, and foundational data science and machine learning concepts, complemented by strong soft skills such as collaboration, communication, critical thinking, problem-solving, and a commitment to continuous learning and adaptability.

REFERENCES

Automation in Clinical Trial Statistical Programming: A Structured Review of TLF Generation, Validation Frameworks, and AI/ML Integration (2020–2025). Jaime Yan, Jason Zhang, Tingting Tian.

Upputuri, V. (2025). Leveraging AI/ML in Statistical Programming: Enhancing Efficiency, Compliance, and Insights in Clinical Trials. International Journal of Research in Computer Applications and Information Technology.

PHUSE Paper PP01 The Evolving Role of Statistical Programmers in the Pharmaceutical Industry: Impacts of Artificial Intelligence.

https://pharmaverse.org/

CONTACT INFORMATION

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

Author Name: Manoj Kumar Maripally
Principal Statistical Programmer, Cytel Inc.
Email: manojkumar.maripally@cytel.com
Website: www.cytel.com