Paper ML03

AutoProgramming: Agents Are All You Need

Chengxin Li, AutoCheng Clinical Data Services LLC

David Li, AutoCheng Clinical Data Services LLC

ABSTRACT

Statistical programming is resource-intensive and largely standardized with CDISC across the Biopharma industry, but programming codes not publicly shared with sponsor-owned intellectual property, limiting Large Language Model (LLM) training. While current LLMs struggle to produce fully reliable, production-ready code, they excel in interpreting and processing documentation.

To bridge this gap, this paper introduces a transitional multi-agent framework that integrates **LLM-based AI Agents** with **SAS Macro Agents** to enable scalable and reliable programming automation.

In this framework, AI Agents interpret study documents to generate metadata that configures Macro Agent execution. For example, when provided with table shells and ADaM specifications, AI Agents extract **Analysis Results Metadata (ARM)** and display features, which Macro Agents then use to produce tables and executable SAS code. Similarly, using study metadata from raw data (EDC/eDT) and the study CRF document, AI Agents can draft mapping specifications, which the **autoSDTM Macro Agents** employ to generate SDTM domains and corresponding SAS programs. The framework can further extend to **autoADaM** workflows such as for safety dataset generation.

Implementing AI Agents requires new skillsets like **prompt engineering** and **agent orchestration** for programmers, while Macro Agents leverage the developed SAS macro tools. This game-changing approach offers a production-focused and scalable pathway toward programming automation, redefining the programmer's role in the emerging era of AI-driven clinical data science.

INTRODUCTION

AI and Machine Learning

Artificial Intelligence (AI) is transforming modern industry at a pace comparable to the Internet Revolution of the 1990s. What once relied on manual coding and deterministic rules (e.g., *if–then–else*) is now augmented by systems that can learn, reason, and interact with humans. At the core of this transformation lies **Machine Learning (ML)**—computers learning patterns from data rather than being explicitly programmed. Within ML, **Deep Learning (DL)** employs multi-layered neural networks to recognize complex relationships, enabling capabilities such as natural-language understanding, predictive analytics, and decision support.

A major breakthrough in recent years is the rise of **Large Language Models (LLMs)**. Trained on terabytes of text and parameterized by billions of weights, these models predict the next token in a sequence and, through scale, demonstrate emergent reasoning abilities. LLMs such as *ChatGPT, Gemini, Claude, Copilot,* and *DeepSeek* are now widely adopted across domains for writing, coding, summarization, and analysis. Their versatility has made advanced AI accessible even to non-technical professionals.

Building upon LLMs, **AI Agents**—also known as *Agentic AI*—represent the next leap forward. Rather than simply generating text on demand, agents can plan, reason, and act within defined workflows. They integrate LLMs with external tools, APIs, and memory modules to perform multi-step tasks autonomously. The year **2025 has been called "The Year of AI Agents,"** marking a shift from passive assistants to active digital collaborators capable of automating routine professional work.

At the same time, an expanding ecosystem of affordable cloud-based AI tools allows organizations to harness these capabilities without deep technical expertise. Many of today's popular systems—such as *ChatGPT, Gemini, Copilot, Claude,* and *DeepSeek*—are available via web or enterprise platforms, making intelligent automation widely accessible.

It is important to clarify, however, that end users cannot train these large models directly, nor do LLMs use individual user data for model training. The underlying models are maintained and improved solely by their developers, based on aggregated, anonymized feedback and ongoing research. This design ensures that professionals can safely leverage AI capabilities without risking the exposure of proprietary or confidential information.

In summary, the current generation of AI tools offers an unprecedented opportunity for productivity and innovation—with safety boundaries preserved through centralized model development and responsible deployment.

Statistical Programming

Within clinical development, statistical programming plays a central role in end-to-end data processing and analysis, transforming raw study data into regulatory-compliant outputs. These workflows operate under strict Standard Operating Procedures (SOPs) and Work Instructions (WIs) to ensure data integrity, traceability, and reproducibility.

Across companies, programming activities are remarkably similar—repetitively reading and transforming data, producing CDISC-compliant SDTM and ADaM datasets, generating Tables, Listings, and Figures (TLFs), and maintaining extensive documentation. **SAS** remains the dominant programming tool, with **R** gaining traction and **Python** emerging in early adoption stages.

Documentation—including specifications, *define.xml* files, Analysis Results Metadata (ARM), Reviewer Guides (cSDRG and ADRG), and annotated CRFs—often consumes nearly as much time as coding itself. Maintaining consistency among these deliverables is a persistent challenge; consequently, AI and ML offer a promising pathway to improve efficiency, quality, and consistency in Biopharma programming.

CHALLENGES ON APPLICATIONS

While Artificial Intelligence (AI) and Machine Learning (ML) continue to advance rapidly, their direct application to statistical programming in the Biopharma industry faces two major barriers: **security concerns** and **business model limitations**.

Security Concerns

The most immediate issue involves data security and compliance. Current apprehension resembles the skepticism that once surrounded the transition from local servers to cloud infrastructure. Pharmaceutical and clinical-trial data are highly sensitive, governed by strict frameworks such as 21 CFR Part 11 and other global data-protection standards.

Most commercial AI tools are primarily web-based or embedded within enterprise software suites, meaning user prompts are transmitted to external servers for processing. This architecture raises valid concerns about whether confidential metadata, specifications, or documents could be inadvertently exposed to external systems.

Nonetheless, industry adoption trends suggest that these security challenges will lessen over time. The growing integration of AI into enterprise environments and browsers shows that vendors are rapidly enhancing security, compliance, and audit controls. As organizations increasingly “embrace AI,” governance frameworks will mature—just as they

did during the early cloud migration—eventually making secure AI environments a standard part of validated infrastructures.

Business Model

A second challenge stems from the business model underlying BioPharma industry. Although statistical programming practices are similar across companies, the actual codebases are proprietary and represent valuable intellectual property (IP). Because these programs embed sponsor-specific standards, algorithms, and domain knowledge, they cannot be publicly shared. Consequently, AI vendors lack access to authentic, production-grade SAS programs or CDISC-compliant code to train their models effectively.

As a result, current LLMs excel at interpreting documentation and drafting pseudo-code, but the outputs are often not production-ready—and may even be incompatible with standard code reuse practices such as copy/paste or updates from existing validated programs.

Because such proprietary data will likely remain private, meaningful improvements in the reliability of AI-generated programming may take several years. Until then, programmers can effectively use LLMs for documentation support, explanation, and metadata extraction, but not yet for the fully automated generation of validated production code.

Summary

In conclusion, security governance and restricted training data remain the two principal barriers to adopting AI/ML in statistical programming. Both areas are expected to improve as technology evolves and as vendors and sponsors collaborate on compliant solutions. In the near term, however, AI's role will remain supportive rather than fully autonomous, complementing—not replacing—human programmers in regulated production environments.

SOLUTION-ALL IN AGENTS

Although current Large Language Models (LLMs) face challenges in generating reliable, production-ready code, their ability to process, interpret, and structure documentation provides tremendous potential for workflow automation.

To leverage these strengths while maintaining high quality and high efficiency, a transitional multi-agent framework has been proposed and referred to as **All in Agents**.

This approach integrates AI Agents (LLM-based) with Macro Agents (SAS macro–based) to form an end-to-end automation system with minimal manual effort in the operations.

Framework Overview

The “All in Agents” framework establishes a collaborative interaction between two complementary agent types:

- Applied AI Agents (LLM-based): Responsible for interpreting study documents, extracting metadata, and preparing configuration parameters.
- Macro Agents (SAS macro–based): Responsible for executing deterministic programming steps, using the parameters generated by AI Agents to produce outputs including plain executable SAS codes.

Together, these components form a multi-agent ecosystem that combines the reasoning ability of AI with the reliability of rule-based SAS macros. This hybrid design preserves previous automation investments while introducing AI-driven intelligence for documentation-heavy processes.

AI Agents

Applied AI Agents serve as the *interpretive layer* between unstructured documentation and structured automation. They utilize LLMs to read, understand, and summarize information from study-level documents—such as metadata of raw data, CRF, table shells, ADaM specifications, and Statistical Analysis Plans (SAPs)—then convert those insights into metadata that can drive downstream macros.

For examples:

- When provided with study metadata from raw-data (EDC/eDT) and an annotated Case Report Form (CRF), the Agent can automatically draft SDTM mapping specifications, which aligned with autoSDTM Macro Agent parameters
- Given table shells and ADaM specifications, an AI Agent extracts Analysis Results Metadata (ARM) and key layout information, which aligned with autoTable Macro Agent parameters

These AI Agents act as the cognitive engine that prepares inputs for automated execution. Through prompt engineering, Agents can follow standardized reasoning sequences—planning, checking consistency, and outputting parameters in structured formats.

To enhance both accuracy and context control, the framework incorporates Retrieval-Augmented Generation (RAG). RAG allows the AI Agent to access relevant documents or templates during response generation.

Importantly, the LLM is restricted from accessing raw clinical data; it only processes textual documentation and metadata, ensuring data privacy and regulatory safety.

Macro Agents

Macro Agents are the *execution layer* of the framework. They consist of developed SAS macros that perform deterministic tasks using the metadata supplied by AI Agents.

Examples include:

- **autoSDTM:** Driven by SDTMIG metadata and mapping specifications, automatically creates SDTM domains and corresponding plain executable SAS codes.
- **autoTable:** Produces analysis outputs including plain executable and submission-ready SAS codes, based on ARM and key layout configurations.

Just for one instance, there are four autoTable Agents developed and applied to perform any descriptive analyses in a study: **ADSLEVL, BDSSTAT, BDSSHIFT, and OCCFREQ**. The corresponding outputs are listed in Table 1 below.

Table 1. Macro Agent Outputs

Seq	Item	Note
1	Table	One or more table output with formats of RTF, PDF, XLSX
2	SAS Codes	Plain executable SAS codes dynamically resolved from the macro agent call. If click RUN, it can produce the same table
3	RDS	Final results dataset used for qc, or any other layout from on
4	ARMDS	Aggregated dataset of table specifications to review and facilitate ARM define-xml generation
5	Log file	Macro Agent execution logs

Oversight and Governance

Even with automation, user oversight remains essential. Users review and approve all AI-generated parameters before execution. This human-in-the-loop design ensures that while automation accelerates documentation and code generation, final accountability and quality control still stay firmly with the programming team.

Summary

The **All in Agents** framework provides a practical, production-focused path toward programming automation. By combining AI Agents for intelligent interpretation with Macro Agents for execution, it offers a realistic bridge between today's documentation-heavy programming practices and tomorrow's intelligent, self-orchestrated workflows.

Below figure 1 illustrates one example of the All in Agents model for autoTable.

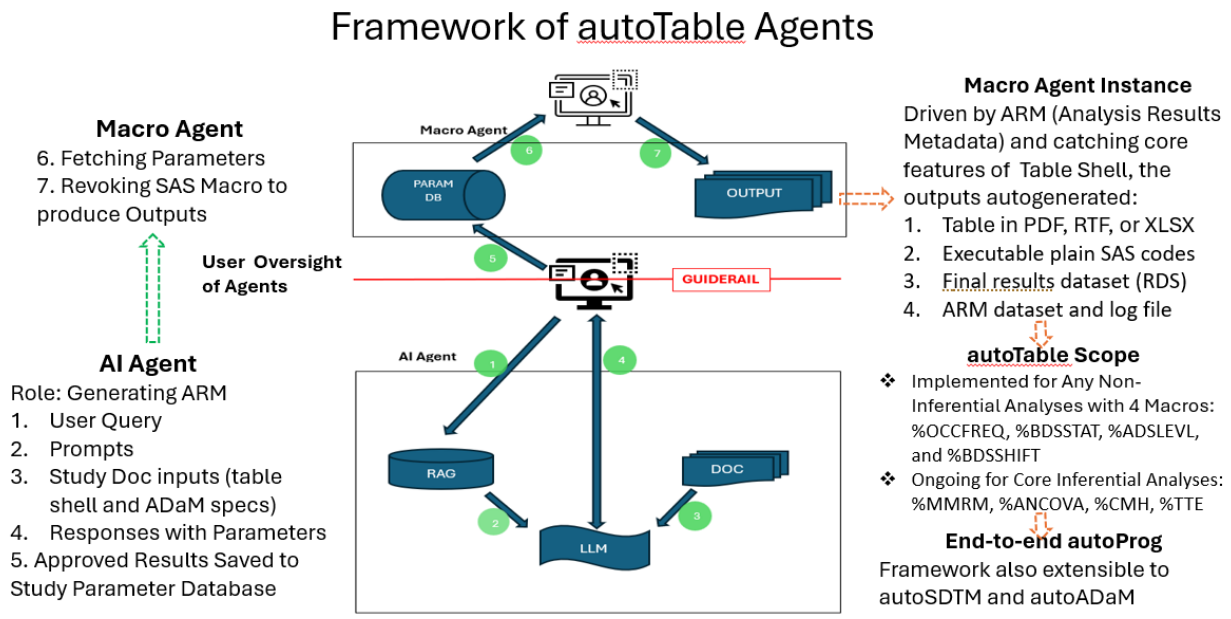

Figure 1 Framework of autoTable Agents

This layered approach allows end-to-end automation. By keeping AI interpretation separate from deterministic execution, the system ensures transparency and reproducibility.

EMERGING SKILLS

The introduction of AI Agents and Macro Agents into clinical-programming workflows does more than automate tasks—it reshapes the skillset required of programmers. To implement and maintain multi-agent systems effectively, professionals must develop competencies that blend statistical-programming expertise with AI literacy, communication design, and process integration.

AI/ML Fundamentals

As AI technologies become embedded in day-to-day operations, programmers need at least a working understanding of AI and Machine-Learning fundamentals. Key concepts may include neural-network structure, Deep Learning (DL) mechanisms, and how LLMs process tokens and context windows.

More advanced understanding involves Agentic AI components—such as reasoning chains, planning modules, and multi-component orchestration frameworks (e.g., LangChain). Knowledge of Retrieval-Augmented Generation (RAG), Memory, and multi-component

planning (MCP/A2A) also helps programmers appreciate how AI Agents combine reasoning with contextual retrieval to ensure reproducible, auditable outcomes.

Most LLM providers now offer Agent Development Kits or low-code tools that require minimal programming expertise with natural language descriptions. Even without writing deep-learning code, statistical programmers will benefit from knowing how these systems think and how to guide them safely within regulated environments.

Prompt and Context Engineering

A pivotal new capability in the age of intelligent automation is Prompt Engineering, complemented by the emerging concept of Context Engineering. Together, they determine how effectively an AI Agent interprets user intent and produces reliable, high-quality outputs.

Prompt Engineering focuses on *how* instructions are communicated to the AI—using clear roles, structured tasks, logical sequencing, and examples to achieve consistent results. Context Engineering focuses on *what information* is provided around those prompts—defining the background, reference materials, and metadata that the Agent can access to reason accurately. In essence, prompts guide intent, while context guides content.

Effective prompt and context design specify:

- the role of the AI Agent (e.g., *"clinical-programming assistant"*),
- the task scope and constraints (such as output format),
- step-by-step reasoning guidance, and
- relevant contextual references—including prior specifications, standard templates, or controlled terminology retrieved via RAG.

In statistical programming, strong prompt and context engineering allow AI Agents to:

- Generate draft ADaM or SDTM specifications from SAP or CRF documents.
- Create and align ARM parameters for Macro Agents.
- Validate metadata consistency across deliverables such as specifications, annotations, and reviewer guides.

By mastering both prompting and contextual design, programmers move beyond traditional coding to orchestrating intelligent, context-aware workflows. These dual skill sets define effective communication with AI tools to ensure AI outputs remain accurate, explainable, and auditable—key requirements in regulated Biopharma environments.

Process Integration and Governance

AI Agents are not meant to replace established infrastructure but to augment it—operating within current business models, audit trails, and SOP environments. Routine, well-structured workflows can be delegated to AI Agents that plan, reason, and act autonomously, while programmers maintain oversight of the logic and outputs.

To address security and compliance, all external AI interactions are restricted to study documents and metadata, never patient-level data. This approach allows organizations to achieve immediate efficiency gains without major process disruption or regulatory risk.

Summary

In summary, the adoption of AI Agents in statistical programming introduces a new skill paradigm. In this evolving environment, statistical programmers must continuously reskill—combining traditional data standards expertise with modern AI fluency. With appropriate training and process safeguards, these skills position the Biopharma workforce to harness automation responsibly while maintaining high quality and high efficiency.

DISCUSSION

The introduction of AI Agents and Macro Agents marks a pivotal transition in statistical programming—from manual, code-intensive processes toward intelligent automation guided by metadata and documentation. While the framework presented here focuses on practical implementation, it also opens new directions for research, organizational change, and workforce development within the Biopharma analytics community.

Future Work Direction

AI Agents have the potential to be a transformative change in clinical programming. Future development will focus on expanding automation beyond current proof-of-concept modules to a fully integrated, multi-agent ecosystem supporting SDTM, ADaM, and TLF workflows. Key directions include:

- **Digitalization of costing and resource models**, where agent throughput and validation efficiency replace manual coding hours as the primary productivity measure.
- **Redesign of reporting frameworks**, evolving from single-script pipelines to orchestrated agent systems that coordinate document interpretation, metadata management, and SAS/R/Python execution.

- **Migration toward open ecosystems**, where today's *LLM + SAS macro* architecture gradually transitions into *LLM + R/Python package* frameworks as those languages achieve comparable regulatory maturity.
- **Macro Agent evolution**, in which current SAS-based macros may eventually be upgraded into fully AI-enabled tools once models become capable of generating reliable, validated programs autonomously.

These directions emphasize that automation is not a single project but a progressive evolution—each phase building upon proven compliance and data-quality foundations.

Roles of the Programmer

Even in an era of advanced AI, the programmer still remains central to successful clinical data analysis delivery. High-quality output continues to depend on the harmony between people, technology, and process. With AI Agents relieving programmers of routine, repetitive work, the human role shifts toward higher-value activities.

In this All in Agents model, programmers evolve into AI orchestrators—professionals who combine domain expertise, technical fluency, and strategic judgment to guide intelligent systems responsibly. Rather than replacing human skill, AI extends its reach, allowing programmers to focus on creativity, oversight, and innovation.

CONCLUSION

The integration of AI Agents and Macro Agents presents a practical and scalable path toward reliable programming automation in the Biopharma industry. While current Large Language Models (LLMs) are not yet capable of generating fully validated production code, their strength in interpreting and structuring documentation makes them powerful enablers of efficiency and quality.

By coupling these interpretive capabilities with deterministic developed SAS Macro Agents, the proposed "All in Agents" framework delivers a balanced approach—intelligence where flexibility is needed, and automation where precision is critical.

This framework preserves data integrity and regulatory compliance while significantly reducing repetitive programming and documentation efforts. At the same time, it redefines the programmer's role: from manual coder to AI-enabled orchestrator responsible for guiding intelligent agents, validating outputs, and innovating processes. As the Biopharma industry continues its digital transformation, such hybrid AI frameworks will become

essential to managing growing data volumes, accelerating reporting timelines, and ensuring reproducible analytics.

In essence, Agents are all you need—not to replace programmers, but to empower them. Through responsible implementation, human oversight, and continuous innovation, AI Agents will transform statistical programming into a more efficient, intelligent, and future-ready discipline.

CONTACT INFORMATION

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

Author Name: Chengxin Li

Company: AutoCheng Clinical Data Services LLC

Email: Chengxin.li@autoclindata.com

David.li@autoclindata.com