From Raw Data to Optimized Models: Reinforcement Learning–Driven AI Agents to Automate Biopharmaceutical Modeling Workflows

Nima Sammaknejad

Data Science Technical Lead

nima.sammaknejad@gmail.com

https://www.linkedin.com/in/nima-sammaknejad-ph-d-12743256/

Abstract

Reinforcement Learning (RL) has recently emerged as a promising approach to automate complex decisions in machine-learning pipelines, particularly for data preprocessing and model selection. In biopharmaceutical modeling workflows such as spectroscopic analysis and batch process modeling, these decisions are typically performed manually, relying on expert judgment and extensive trial-and-error. Prior work has demonstrated that RL can effectively learn sequential data-cleaning or modeling actions, but applications to end-to-end biopharmaceutical data science workflows remain limited.

In this article, an RL-driven autonomous agent is developed to optimize spectroscopic preprocessing decisions using a Markov Decision Process (MDP) formulation integrated with LangGraph. The agent learns an optimal sequence of spectral transformations including smoothing, derivative operations and model optimization that minimizes cross-validated prediction error. Using a public Near-Infrared (NIR) dataset, the proposed agent consistently discovers preprocessing chains that outperform baseline and manually developed models. This work demonstrates the feasibility of embedding RL-based decision-making within structured graph frameworks to enable reproducible, data-driven and fully automated modeling workflows.

1. INTRODUCTION

Data pre-processing decisions, e.g., cleaning signals, choosing the right transformations and feature engineering are among critical bottlenecks in data science workflows. These steps normally include countless subjective decisions and trial and error before achieving the desired results. These challenges extend across biopharmaceutical modeling activities, e.g., batch process modeling, spectroscopic data modeling, etc. where similar complex preprocessing and modeling chains are required ([1], [2], [7]). Recent studies such as ReClean illustrate that data-cleaning pipelines involve many sequential decisions that can influence the outcome [3]. DeepLine expands this challenge to full machine learning pipelines involving interdependent preprocessing and modeling decisions [4]. In both ReClean and Deepline, it is demonstrated that Reinforcement Learning (RL) can be employed to automate manual data preparation or pipeline decisions. In such cases, the RL agent selects the optimal data preprocessing step or pipeline decision, which ultimately results in superior model performance ([3], [4]). More recently, agent-oriented frameworks like Intelligent Spark Agents highlight the feasibility of automating such end-to-end data science workflows using structured graph-based ecosystems such as LangGraph [5].

In biopharmaceutical industry, multi-step modeling workflows exist across chromatography, batch/continuous process modeling and Process Analytical Technology (PAT)-driven prediction tasks. Near-Infrared (NIR) and Raman spectroscopy models are among common PAT technologies to estimate drug properties using spectral data as model input. In standard practices, preprocessed spectra are regressed against the attribute of interest, and the resulting model is used when direct measurement of the attribute is not possible. Preprocessing of spectroscopic data requires a long sequence of decisions including wavenumber selection, Unit-Variance (UV) or Standard Normal Variate (SNV) scaling, Savitzky–Golay (SG) smoothing, first- or second-derivative transformations, feature interactions, etc. Often, extensive experimentation is required before producing a reliable model. Recent developments in spectroscopic modeling workflows show how these decisions form a large combinatorial search space that grows rapidly with the number of features (wavenumbers) ([6], [7]).

Instead of brute-force exploration of this search space, which is impractical, reinforcement learning provides a structured sequential decision-making framework for learning optimal transformations that lead to superior model performance. These learned RL-based decisions can then be embedded into structured graph frameworks, e.g., LangGraph, to form autonomous AI agents capable of guiding users directly toward optimized spectroscopic models. In this article, the RL-based decision-making integrated with LangGraph is leveraged to develop an autonomous AI agent that performs the optimal preprocessing sequence and delivers the final model.

2. PROBLEM STATEMENT

As illustrated in Figure 1, spectroscopic modeling data workflow is normally an iterative process that spans multiple stages. This includes data collection and aggregation, wavenumber selection, spectral processing, outlier detection, feature engineering and model evaluation. Decisions made early in the pipeline can propagate through the entire workflow. Small choices such as SNV scaling versus unit variance, selecting appropriate Savitzky–Golay parameters, or choosing the right derivative order can drastically impact the final model ([6], [7]). Comparable decisions exist across other biopharmaceutical modeling tasks such as cell-culture monitoring, purification process modeling and multivariate PAT analysis ([1], [2]).

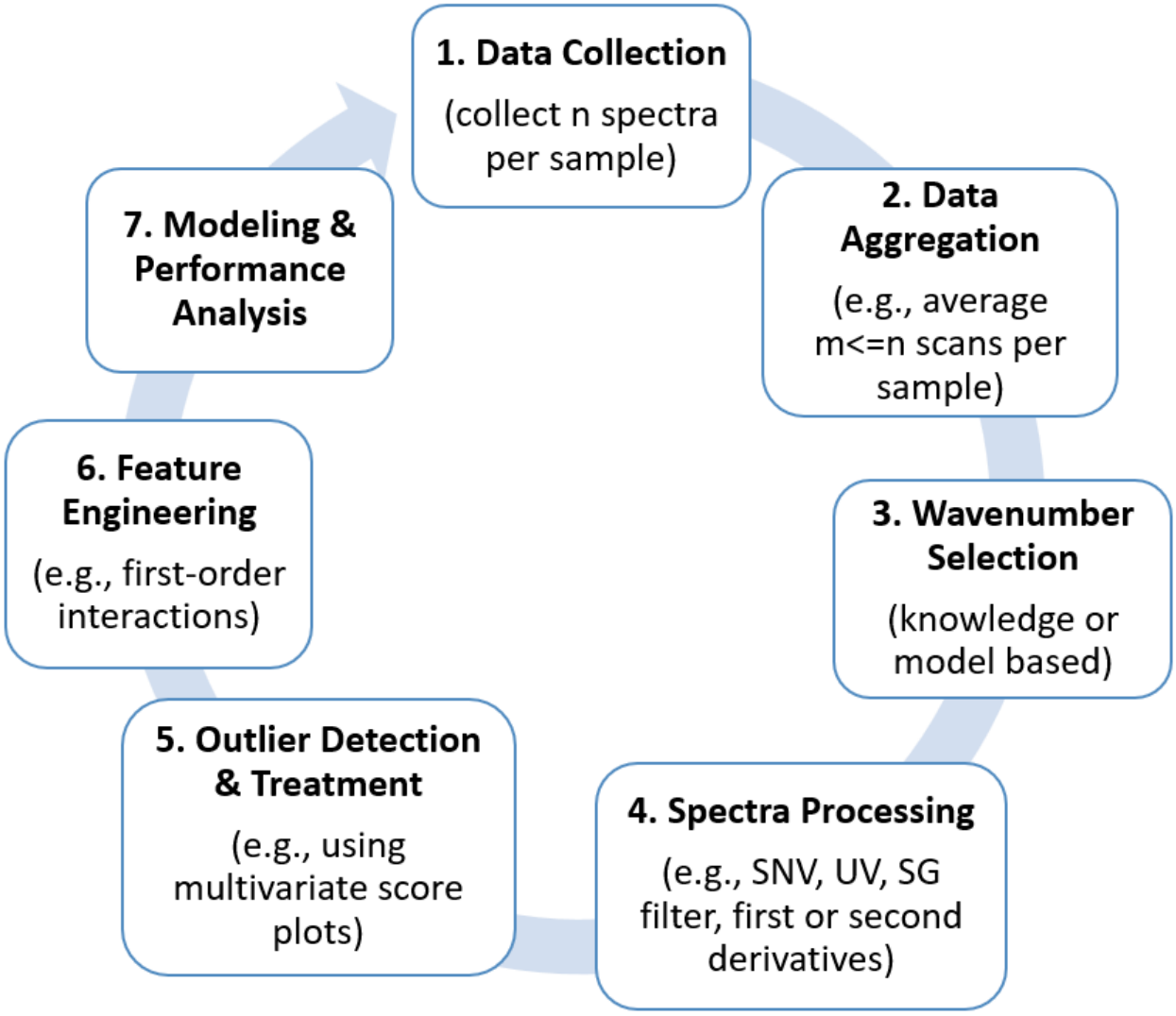

Figure 1. Spectroscopic modeling data workflow

These modeling workflows, including spectroscopic and other biopharmaceutical modalities, are typically performed manually via trial and error. Thus, expert judgment can significantly impact the results, and different analysts may produce different results from the same dataset. This dependency on trial-and-error and expert exploration makes preprocessing both time-consuming and highly inconsistent across teams and projects. As a result, preprocessing and model selection remain some of the most difficult, subjective, and variable components of biopharmaceutical data science, including spectroscopic workflows.

The existing methods are limited in the sense that they cannot infer or optimize the sequence of preprocessing steps. They depend on expert choices or predefined pipelines that cannot

generalize across datasets. As a result, preprocessing remains a manual, expert-driven process, preventing automation and limiting both speed and consistency. The approach proposed in this article follows a similar direction to recent advances such as ReClean, DeepLine, and Intelligent Spark Agents. Reinforcement learning is used to learn an optimal policy that selects the best sequence of preprocessing actions, and this policy is integrated into a graph-based agentic AI framework (LangGraph) to automate the end-to-end spectroscopic data workflow.

3. METHOD

3.1 Reinforcement Learning

The model-based reinforcement learning (RL) framework used in this article is formulated as a Markov Decision Process (MDP) to achieve optimal sequential actions. An MDP contains state space (S), action space (A), transition probabilities $p(s_{t+1}|s_t, a_t)$ and reward function R [8]. The objective is to learn the policy $\pi(s)$ that maximizes the discounted return as in Equation 1 [8].

$$\boldsymbol{G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}} \qquad \text{Equation 1}$$

where γ is the discount factor, t denotes the current time step and k indexes future steps in the discounted return.

Under a given policy π, the state-value function is as in Equation **2**.

$$\boldsymbol{\nu_\pi(s) = E_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right]} \qquad \text{Equation 2}$$

Similarly, the action-value function can be written as in Equation **3**.

$$\boldsymbol{q_\pi(s, a) = E_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a \right]} \qquad \text{Equation 3}$$

where $\nu_\pi(s)$ is the discounted expected return over all trajectories generated by following policy π from state s, and $q_\pi(s, a)$ is the expected return when starting in state s, taking action a, and then following policy π afterward [8].

The optimal action-value function q^* satisfies the Bellman optimality relation. Following the optimal policy π^*, q^* equals the expected one-step reward r_{t+1} plus the discounted optimal future state value $v^*(s_{t+1})$ as shown in Equation **4** [9].

$$\boldsymbol{q^*(s, a) = E[r_{t+1} + \gamma \nu^*(s_{t+1}) | \, s_t = s, a_t = a \,]} \qquad \text{Equation 4}$$

Within the model-based dynamic programming framework, two common approaches are used to determine the optimal policy in an MDP: value iteration and policy iteration. Value iteration

is derived from the Bellman optimality update in Equation **4**. It includes a recursive maximization step to evaluate and improve the policy, driving the state-value function ν (defined in Equation **2**) toward its optimal value ν^*. Policy iteration alternates between policy evaluation and improvement to achieve the optimal policy [9].

Regardless of the method (value or policy iteration), the optimal policy π^* is obtained by selecting the action a^* that maximizes the Bellman equation as in Equation **5** [9].

$$\boldsymbol{\pi^*(s) = \arg\max_a q^*(s,a) = a^*} \quad \text{Equation 5}$$

While policy iteration tends to converge in fewer iterations, value iteration is often more practical because it combines evaluation and improvement into a single update. In the current study, value iteration is leveraged.

3.2 Automated RL-Driven Data Workflow

In this study, to illustrate the concept within the broader biopharmaceutical setting, a spectroscopic case study is used. The existing public Near-Infrared (NIR) dataset for fresh peaches to predict the brix values (peach_spectra_brix.csv in reference [10]) is used to demonstrate the concept. In chemometrics, Partial Least Squares (PLS) is normally the baseline approach to solve this regression problem [10].

The spectroscopic preprocessing workflow is automated through a reinforcement-learning agent that applies a learned MDP policy to the spectra. Each spectral preprocessing decision, e.g., SG smoothing, first or second derivative transformation, or PLS model optimization is represented as an action in the MDP. MDP states encode the sequence of transformations applied to the spectra starting from raw data. The reward is given by the negative PLS Cross-Validated (CV) Mean Squared Error (MSE) as in Equation **6**.

$$\boldsymbol{r_{t+1} = -MSE_{CV}(t+1)} \quad \text{Equation 6}$$

Using the learned optimal policy π^*, the agent selects the transformation that is expected to yield the largest expected return defined by the learned policy. With each action, the agent applies the corresponding preprocessing tool to the spectral data, generates a new cache entry representing the transformed spectra, and evaluates the representation using PLS CV-MSE, from which, the reward r_{t+1} is computed. This MDP formulation allows the system to discover and execute an optimal preprocessing chain without manual intervention.

3.3 Agent Orchestration and Execution with LangGraph

The execution of this RL-guided workflow is implemented using a structured LangGraph architecture [11]. The graph consists of an LLM decision node, a tool execution node and an environment node that updates the current MDP state after each tool call. This environment node

differentiates the design from reactive agent frameworks (e.g., ReAct), enabling explicit state transitions tied to the MDP structure rather than free-form reasoning loops. The LangGraph architecture is illustrated in Figure 2.

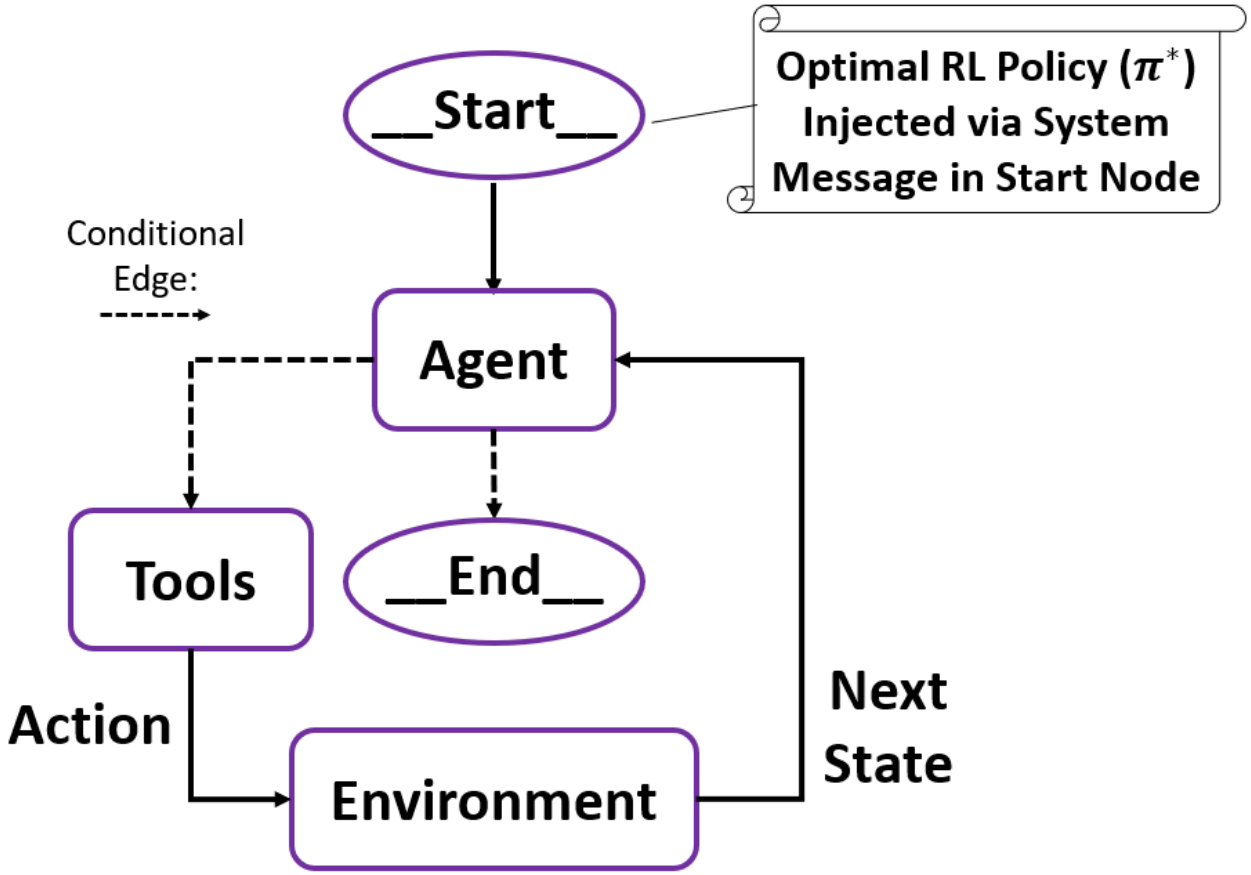

Figure 2. LangGraph architecture of the spectroscopic modeling AI agent

During execution, the agent runs step-by-step on the real data as follows:

1. LLM agent reads the current state and selects the action prescribed by π^*.
2. Agent invokes the corresponding tool.
3. Environment node emits the updated state back into the graph.
4. Loop continues until the terminal state is reached.
5. After termination, the agent provides the final PLS performance metrics to the user.

This structured graph design ensures deterministic compliance with the learned policy while still allowing interactive execution with end users.

4. RESULTS

4.1 Baseline Model

To develop the baseline model, the raw spectral data (peach_spectra_brix.csv) from reference [10] is imported and illustrated in Figure 3. Wavenumbers on the x-axis (with magnitude on the y-axis) serve as features to predict the brix values (target).

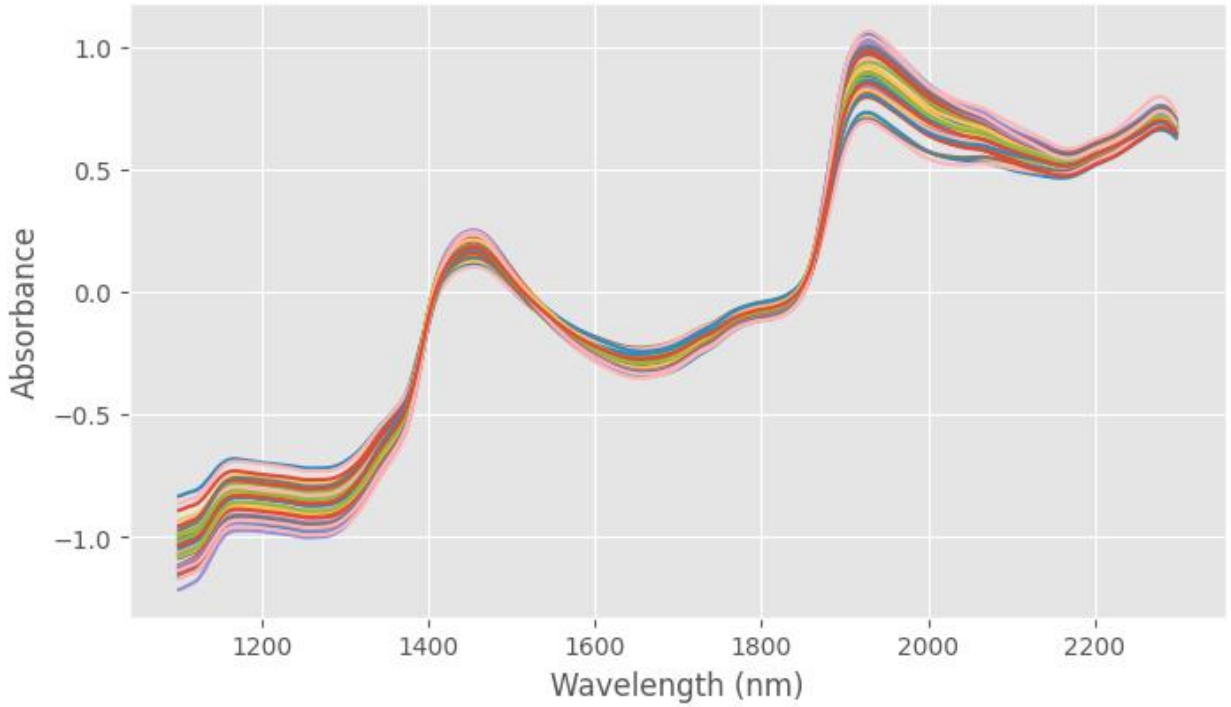

Figure 3. Raw spectral data from reference [10]

Figure 4 illustrates that it is possible to run a k-fold cross-validation to evaluate cross-validation performance. In summary, Figure 4 shows that 6 PLS principal components result in the optimal (minimum) MSE CV equal to 2.98.

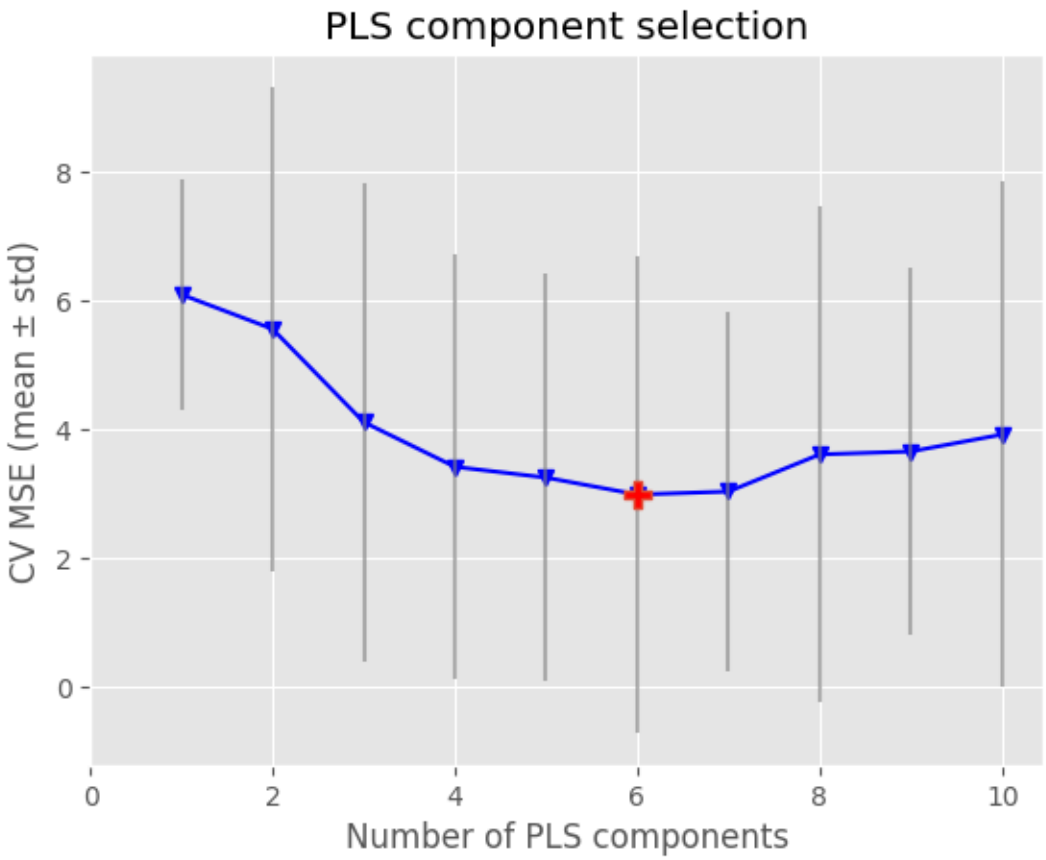

Figure 4. k-fold cross validation on raw spectral data in Figure 3

4.2 Pre-processing Impact on Model Performance

As explained in Figure 1, different pre-processing methods may be applied to extract additional features and improve model performance. For illustration purposes, first derivative of the spectra, which is a common pre-processing method, is illustrated in Figure 5.

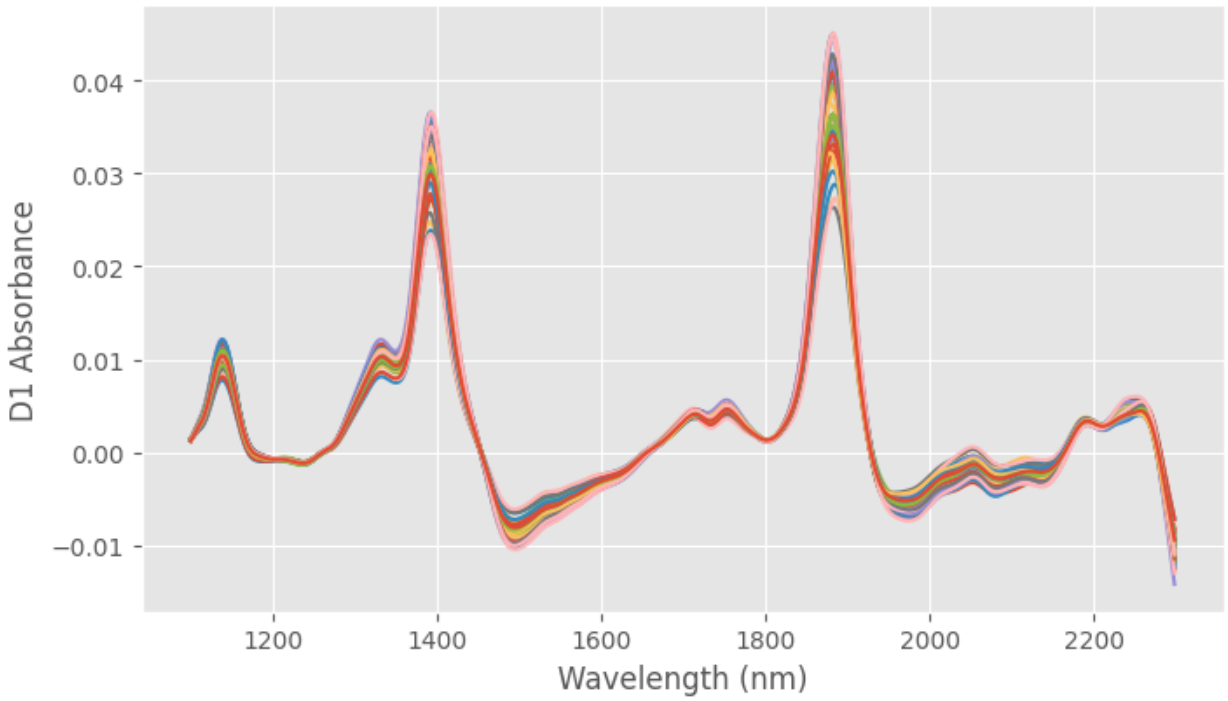

Figure 5. First-derivative of the spectral data in Figure 3

Cross-validation modeling outcome is illustrated in Figure 6. The results indicate that with half number of principal components, that is, 3 versus 6 in Figure 4, slightly better model performance with MSE CV equal to 2.808 is achieved.

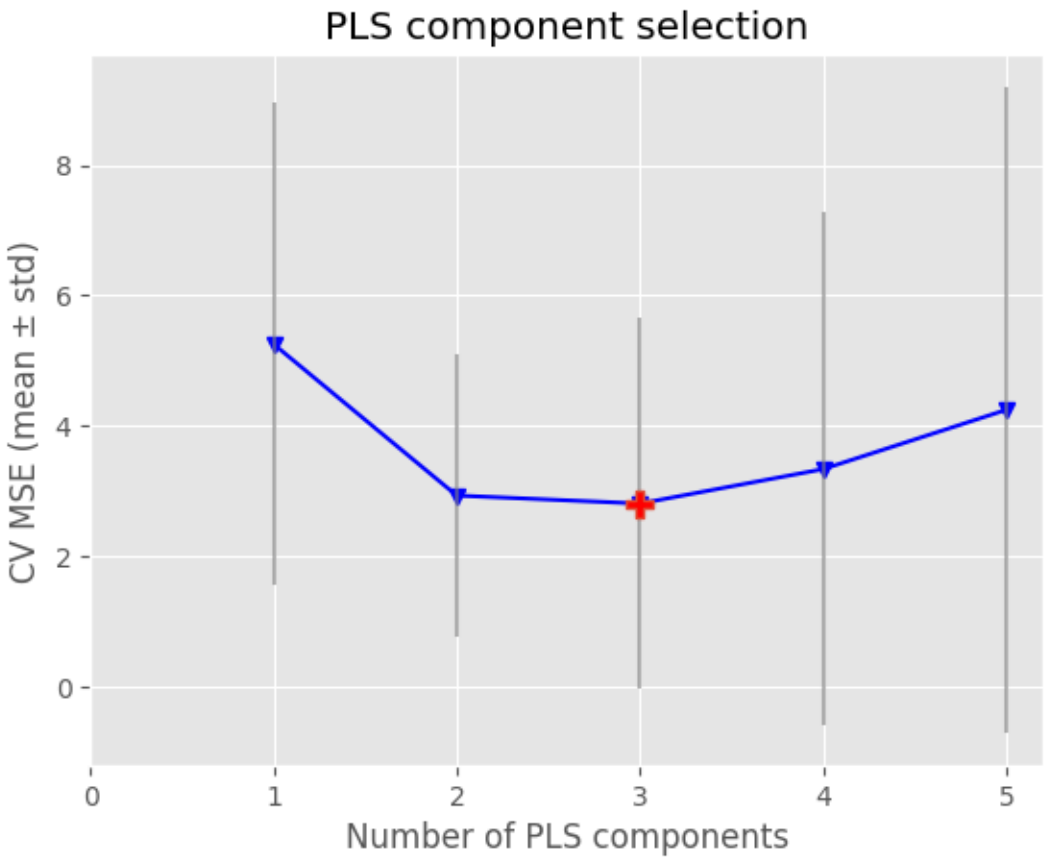

Figure 6. k-fold cross validation on first-derivative spectral data in Figure 5

4.3 RL-Driven AI Agent Performance

As explained in Figure 1, it is not practical to conduct brute-force search to evaluate all pre-processing combinations. For illustration purposes, three types of pre-processing including SG smoothing, first and second derivative are considered in this study. It is possible to have combination of pre-processing steps, for example, first derivative followed by SG smoothing. In this study, the MDP includes up to two sequential preprocessing steps. States, actions, MDP and LangGraph architecture are designed as explained in the Method section.

The summary outcome of AI agent's progress to apply pre-processing steps and report out the optimal model performance is illustrated in Figure 7. Following the optimal policy π^*, the agent applies two consecutive first-derivative pre-processing steps. The final model with MSE CV equal to 2.5335 demonstrates significant improvement in comparison to the manual methods in sections 4.1 and 4.2 and the outcome in reference [12], where fewer PLS components achieve superior performance.

Human Message: Start episode in **S_RAW** using cache_id *nir_raw*. **AI Message:** Policy π^* selects **A_Preproc_FirstDeriv**. **Tool Message:** First derivative applied → new cache_id → **S_RAW_FirstDeriv.**
AI Message: Policy selects **A_Preproc_FirstDeriv** again. **Tool Message:** First derivative applied again → **S_RAW_FirstDeriv_FirstDeriv**.
AI Message: Policy selects **A_Optimize_Model**. **Tool Message:** PLS CV executed → **cv_mse = 2.5335**, reward = –2.5335 (n = 50, features = 600)
Human Message: CURRENT_MDP_STATE = **S_TERMINAL** **AI Message:** Calls terminate_episode and returns final metrics.

Figure 7. Summary outcome of AI agent's progress to apply pre-processing steps and report out the optimal model performance

5. CONCLUSIONS

This study demonstrates that reinforcement learning can successfully guide an AI agent to automate complex preprocessing steps within biopharmaceutical modeling workflows and achieve improved regression performance over manual methods. By learning an optimal sequence of transformations, the RL-driven agent consistently produces superior PLS models with fewer components and reduced trial-and-error effort. This structured approach enables reproducible, data-driven preprocessing decisions and reduces reliance on expert intuition. The same framework can be readily extended to other chemometric, multivariate or batch modeling tasks offering a scalable foundation for intelligent workflow automation in the biopharmaceutical industry.

6. CODE REPOSITORY

The implementation of the MDP framework, preprocessing tools and LangGraph agent is available here:

GitHub Repository: https://github.com/nimasammaknejad-afk/RL_AI_Agent/tree/main

7. REFERENCES

[1] Sammaknejad, N., Lee, J., Austria, J. M., Duenas, N., Heiba, L., Sridharan, G., Davis, J., & Undey, C. (2025). A scalable deep learning approach for real-time multivariate monitoring of biopharmaceutical processes with no prior product-specific history. Biotechnology and Bioengineering, 122(9), 2333–2352

[2] Sammaknejad, N., Lee, J., Austria, J. M., Duenas, N., Heiba, L., Reed, D., Ephraim, B., Sridharan, G., Justice, J., Davis, J., & Undey, C. (2024). Real-time multivariate statistical monitoring of biopharmaceutical processes with no prior product-specific history. Computers and Chemical Engineering, 189, 108788

[3] Abdelaal, M., Yayak, A. B., Klede, K., Schoning, H. (2024). ReClean: Reinforcement learning for automated data cleaning in machine learning pipelines. DBML@ICDE'2024

[4] Heffetz, Y., Vainshtein, R., Katz, G., Rokach, L. (2020). DeepLine: AutoML tool for pipeline generation using deep reinforcement learning and hierarchical actions filtering. KDD '20: The 26th ACM SIGKDD

[5] Wang, X., & Duan, Y. (2024). Intelligent Spark Agents: A modular LangGraph framework for scalable, visualized, and enhanced big data machine learning workflows. arXiv preprint arXiv:2412.01490

[6] Sammaknejad, N., Saucedo, V. M., Lazzareschi, J. R., Woon, N. T., Sridharan, G., & Ephraim, B. (2024). Real-time automated monitoring and control of ultrafiltration/diafiltration (UF/DF) conditioning and dilution processes (U.S. Patent Application No. US 2024/0189774 A1).

[7] Rashedi, M., Khodabandehlou, H., Wang, T., Demers, M., Tulsyan, A., Garvin, C., & Undey, C. (2024). Integration of just-in-time learning with variational autoencoder for cell culture process monitoring based on Raman spectroscopy. Biotechnology and Bioengineering, 121(7), 2205–2224

[8] Hassanpour, H., Mhaskar, P., & Corbett, B. (2024). A practically implementable reinforcement learning control approach by leveraging offset-free model predictive control. Computers and Chemical Engineering, 181, 108511

[9] Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction (2nd ed.). MIT Press.

[10] GitHub https://github.com/nevernervous78/nirpyresearch

[11] Foundation: Introduction to LangGraph https://academy.langchain.com/courses/intro-to-langgraph

[12] NIRPY Research https://nirpyresearch.com/partial-least-squares-regression-python/