Generating accurate kinetic models via PPO refinement steps

Team Luka Doncic

M. Vukasinovic **N. Jovanovic** **S. Nikolic** **A. Dávid** **M. Vanouek** **L. Cizinsky**

supervised by **Ljubisa Miskovic and Ilias Toumpe**

Abstract

Kinetic models of cellular metabolism require accurate enzyme-specific parameters to predict dynamic cellular responses, yet experimental data are scarce, and high-dimensional parameter spaces challenge traditional approaches. Building on the NES-based RENAISSANCE framework, we introduce **RL-RENAISSANCE**, which frames kinetic parameterization as a sequential decision process and employs Proximal Policy Optimization (PPO) to iteratively refine a 384-dimensional parameter vector over discrete time steps. At each step, the agent proposes parameter updates sampled from a learned Gaussian policy, receiving rewards based on the dominant eigenvalue of the models Jacobian, clipped and transformed to ensure stability and smooth gradients. In benchmarking against RENAISSANCE on *E. coli* metabolism, RL-RENAISSANCE attains a 91% success rate in generating biologically plausible models, thus nearly matching the 95% incidence of the original framework, while reducing sample usage by over 40% and demonstrating rapid convergence within early refinement steps. Analysis of reward dynamics and eigenvalue distributions confirms progressive learning, although PPOs sensitivity to hyperparameters and occasional overshooting of optimal regions suggest different directions for further improvement. Our results highlight the promise of policygradient methods for sampleefficient, adaptive exploration of kinetic parameter spaces, paving the way for more robust and scalable systemsbiology modeling.

1 Introduction

In biological systems, metabolic processes are governed by a vast network of chemical reactions catalyzed by enzymes. These reactions are characterized by kinetic parameters-quantities such as Michaelis constants and turnover rates-which describe how the rates of biochemical reactions depend on metabolite concentrations. Understanding these parameters is critical for predicting how cells respond to environmental changes, genetic modifications, or drug interventions [2].

Kinetic models provide a mathematical framework that explicitly captures the dynamic behavior of metabolism by incorporating mechanistic knowledge of enzymatic reactions. Unlike purely constraint-based approaches, kinetic models link enzyme activity with metabolite levels and reaction rates, allowing for temporal simulations and deeper insight into cellular behavior. However, constructing such models at scale is notoriously difficult [2]. The challenge lies in the scarcity of reliable kinetic parameter values and the high dimensionality of the parameter space, which make traditional modeling approaches computationally expensive and sensitive to overfitting.

Recent work has introduced generative machine learning methods to efficiently parameterize kinetic models by optimizing for biological plausibility and consistency with experimental data [3].

Frameworks such as RENAISSANCE [4] demonstrate that valid kinetic models can be learned without exhaustive data by leveraging neural networks and evolutionary strategies to explore parameter spaces. RENAISSANCE hypothesizes that evolutionary strategies (ES) outperform reinforcement learning (RL) approaches due to their simplicity, ease of parallelization, and independence from backpropagation [4]. However, to the best of our knowledge, no RL-based method for kinetic model generation currently exists to validate or refute this claim. In this work, we propose RL-Renaissance, a reinforcement learning-based alternative, and systematically compare its strengths and limitations against RENAISSANCE.

In RL-Renaissance, we formulate kinetic parameters as states and train a policy to perform stochastic updates based on their quality: making minimal changes when parameters are valid, and guiding updates when they are suboptimal. Similar to RENAISSANCE, we find that careful hyperparameter tuning is critical for achieving high incidence rates. In particular, insufficient tuning leads to instabilitymanifesting as exploding gradientsor premature convergence to suboptimal solutions. Our results show that RL-Renaissance matches the performance of RENAISSANCE in terms of incidence rate while requiring $1.8\times$ fewer samples. However, due to the sequential nature of policy optimization, training is approximately $2.3\times$ slower under identical hardware constraints (single CPU, no GPU).

We summarise our contributions as follows:

- We introduce RL-Renaissance, a reinforcement learning-based alternative to the ES-based RENAISSANCE framework, capable of achieving comparable performance with significantly improved sample efficiency.
- We conduct a detailed empirical analysis of RL-Renaissance, identifying its strengths as well as its limitations, including increased computational cost and hyper-parameter dependence
- We open-source our implementation at `https://github.com/ludekcizinsky/rl-renaissance` and release the trained policy network to facilitate reproducibility and future research in kinetic model generation using reinforcement learning.

2 Related Work

The development of kinetic models of metabolism has historically been hindered by the difficulty of obtaining reliable kinetic parameters and by the complexity of fitting these models to experimental data. Early efforts often relied on manually curated parameters or brute-force sampling approaches, which were limited in scalability and generalizability [2]. To address these challenges, several computational frameworks have been developed to accelerate and automate the construction of kinetic models. One such framework is [7] (Optimization and Risk Analysis of Complex Living Entities), which introduced a probabilistic approach for generating kinetic models by sampling parameter sets consistent with known biological constraints and steady-state data. ORACLE serves as the conceptual foundation for many subsequent advances in the field, including RENAISSANCE. Recent efforts have further improved parameterization efficiency by incorporating machine learning techniques. For instance, the iSCHRUNK [6] framework uses supervised learning to identify feasible regions in the parameter space, relying on decision trees trained on previously validated kinetic models [2]. Similarly, REKINDLE [3] employs generative adversarial networks (GANs) to learn and sample from the distribution of biologically valid parameter sets. However, both methods depend on training data generated using traditional kinetic modeling pipelines.

The RENAISSANCE framework represents a significant advancement by removing this dependence on prior training data [4]. It optimizes generator neural networks using Natural Evolution Strategy (NES) to produce kinetic parameter sets that yield biologically valid modelsspecifically, models with dynamic responses consistent with experimental time constants. This approach has demonstrated strong performance in generating large-scale kinetic models, such as those of E. coli metabolism, while also supporting the integration of sparse experimental data to reduce parameter uncertainty. While RENAISSANCE demonstrates promising results, it adopts NES as its optimization framework, citing advantages such as simplicity, ease of parallelization, and independence from backpropagation [4]. In this work, we challenge this design choice by introducing a RLbased alternative. Our contributions are twofold: (i) we formulate the kinetic parameter estimation problem within a reinforcement learning framework, and (ii) we provide a systematic comparison between our RL-based

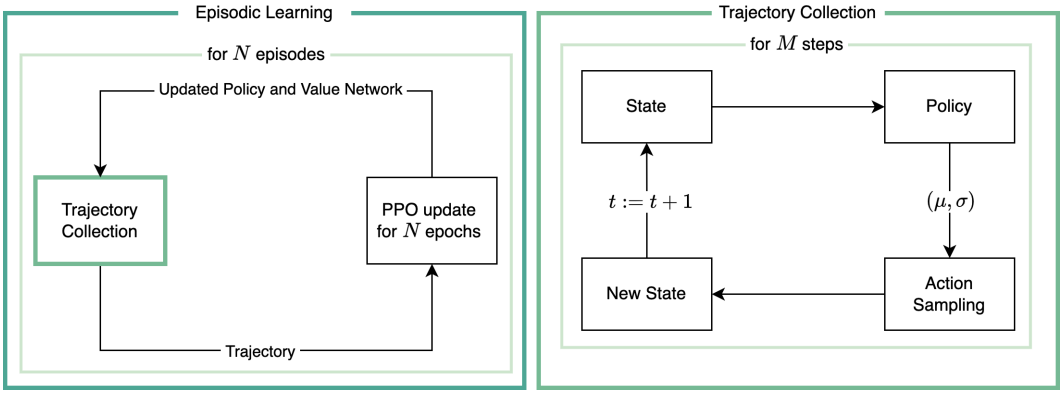


Figure 1: Overview of RL-RENAISSANCE.

approach and the NES-based RENAISSANCE, offering new insights into the trade-offs between these two paradigms.

3 Method Overview

We propose a novel formulation of kinetic parameter generation as a multi-step refinement process guided by reinforcement learning. Specifically, we employ Proximal Policy Optimization (PPO) to iteratively refine a 384-dimensional vector of kinetic parameters, denoted by p, over T discrete steps. The trained policy should therefore learn how to navigate from randomly initiliased constant state to valid kinetic parameters within the T steps.

3.1 Kinetic Parameter Optimization as a Reinforcement Learning Problem

Our method formulates kinetic parameter optimization as a sequential decision-making problem, where an agent incrementally refines a parameter vector $p \in \mathbb{R}^{384}$ over T discrete time steps. The agent is trained using Proximal Policy Optimization (PPO) to generate trajectories $\tau = \{(s_t, a_t, r_t)\}_{t=0}^{T-1}$, with each tuple representing the state, action, and reward at time step t.

At each step, the state s_t captures both the current parameter vector p_t and the timestep t. The action $a_t \in \mathbb{R}^{384}$ is a proposed update to the parameters. The transition from one parameter vector to the next is defined by

$$p_{t+1} = \text{clip}(p_t + \beta a_t, p_{min}, p_{max})$$

where β is a scaling factor that controls the step size. To maintain biological plausibility, parameter values are clipped to remain within predefined feasible bounds after each update. Each state s_t is defined by the current parameter vector p_t.

The policy network $\pi_\theta(a_t \mid s_t)$ defines a conditional distribution over actions given the current state. We parameterize this distribution as a diagonal Gaussian, where the mean $\mu(s_t; \theta)$ and the log standard deviation $\log \sigma(s_t; \theta)$ are outputs of separate heads from a shared neural network trunk:

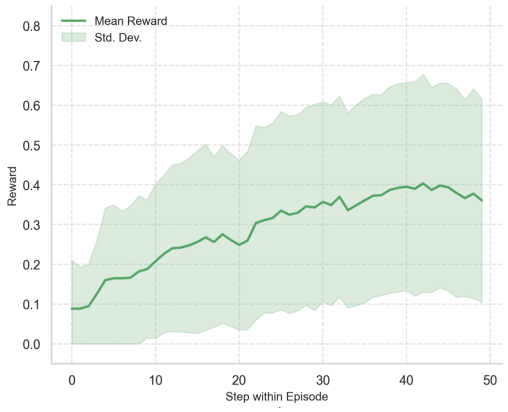
$$a_t \sim \mathcal{N}(\mu(s_t; \theta), \sigma^2 I).$$

This architecture enables both targeted updates (through the learned mean) and exploration (through the learned variance). The stochastic nature of action sampling is essential for effective policy gradient updates and helps avoid local minima in the parameter space.

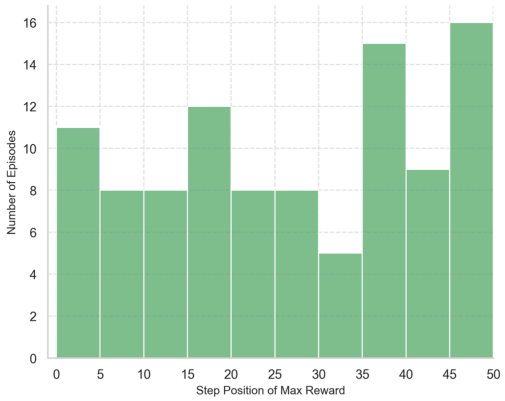
The reward function is designed to promote the generation of dynamically stable kinetic models. We assess stability by computing the dominant eigenvalue $\lambda_{\max}$ of the Jacobian matrix associated with the system's differential equations. A model is considered biologically plausible if $\lambda_{\max}$ falls below a threshold λ_{part}, typically chosen based on empirical time-scale constraints.

To provide a smooth and bounded reward signal, we define the reward as

$$z = \text{clip}(\lambda_{\max} - \lambda_{\text{part}}, -20, +20), \quad r_t = \frac{1}{1 + e^z}.$$

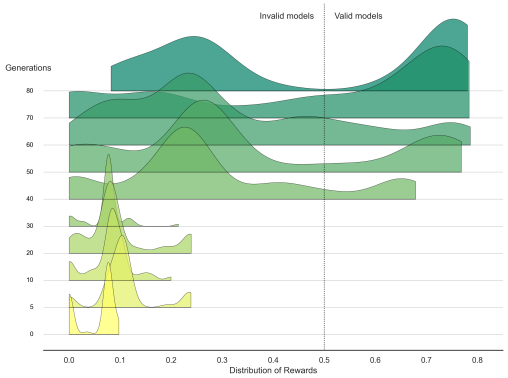


(a) Behaviour of reward function across time steps averaged out across episodes.



(b) Histogram of positions of maximum achieved reward during an episode.

(c) Rewards over episodes for one run.



(d) Evolution of reward distributions across episodes.

Figure 2: Main method training results.

The clipping prevents numerical instability due to extreme eigenvalues, while the sigmoid transformation ensures a continuous and differentiable reward landscape that guides policy improvement.

3.2 Policy Optimization with PPO

Training is conducted using the PPO algorithm, which optimizes a clipped surrogate objective to ensure stable updates. A separate critic network is trained to estimate the value function $V(s_t)$, which is used to compute the advantage estimates $A_t = G_t - V(s_t)$, where G_t denotes the return from time t. We use a high discount factor of $\gamma = 0.99$, as we care about later rewards as much as early rewards. To improve training stability, we scale actions with the factor β, which modulates the aggressiveness of parameter updates. Besides this, careful tuning of learning rates and gradient clipping thresholds is employed for both actor and critic networks to prevent divergence. This reinforcement learning formulation offers a principled and adaptive framework for discovering biologically valid kinetic parameter sets through iterative refinement.

4 Results

4.1 Experimental Setup and Metrics

To evaluate the effectiveness of our reinforcement learning-based refinement strategy, we conducted experiments by running multiple episodes, where each episode corresponds to a full trajectory of T parameter update steps. During an episode, the agent interacts with the environment by sampling actions from the policy network, which are then added (scaled and clipped) to the current parameter

Method	Incidence Rate	Training Wall Time	# Samples Seen
RENAISSANCE	0.95	16 min	$45 \times 20 \times 100 = 90{,}000$
RL-RENAISSANCE	0.91	37 min	$100 \times 50 \times 10 = 50{,}000$

(a) Comparison between RENAISSANCE and RL-RENAISSANCE under a singlecore setup without GPU acceleration.

Ablation	Incidence Rate	Best Step	$\lambda_{\max}$			
			Min	Median	Max	Std
Baseline	0.91	24.9	-3.79	-2.84	182	18.4
200 Episodes	0.59	23.4	-3.87	-2.60	123	20.4
50 Episodes	0.52	26.1	-3.77	-2.52	3960	398.0
Cosine LR Schedule	0.50	30.0	-3.66	-2.41	290	32.3
$\varepsilon_{\text{clip end}} = 0.05$	0.49	27.2	-3.86	-2.16	1920	246.0
$\lambda_{\text{gae}} = 0.95$	0.49	29.6	-3.63	-2.00	13300	1320.0
6 Training Epochs	0.28	27.5	-3.64	-1.58	-0.0023	0.780
Clip Gradient to 1.0	0.27	23.3	-3.77	-1.23	9450	941.0

(b) Ablation study comparing incidence rate and $\lambda_{\max}$ distribution of the produced system, relative to a 100-episode baseline.

vector. At each time step, we compute the reward signal based on the dominant eigenvalue of the system's Jacobian reflecting the stability of the kinetic model at that step.

Throughout the episode, we monitor several key metrics: the maximum reward attained, the minimum reward, and the standard deviation of rewards. These statistics provide insight into the progression and consistency of model refinement. A key goal is for the PPO agent to gradually learn to steer parameter updates toward kinetically stable models, reflected by high reward values and increasingly negative eigenvalues across time.

Over the course of many episodes, we observe how the distribution of dominant eigenvalues shifts. Ideally, this distribution moves below the predefined threshold λ_{part}, indicating improved biological plausibility. In Figure 2d, we visualize this evolution using ridgeline plots to demonstrate how the agent learns to consistently produce more stable kinetic models.

To quantify the overall success of our method, we define the *Incidence Rate* as the proportion of runs (out of N total runs) in which the agent successfully finds a valid set of parameters within the allowed T steps. This metric captures the algorithms reliability in converging to a feasible solution from a random starting point.

4.2 Main results

Our experiments provide several insights into the performance and limitations of the PPO-based approach for kinetic model refinement. First, RL-Renaissance achieves an incidence rate of 91%, closely matching the 95% success rate of the NES-based RENAISSANCE framework, while requiring nearly 20 times fewer samples (Table 1a). This result suggests that reinforcement learning, despite its algorithmic complexity, can offer significantly improved sample efficiency when navigating high-dimensional parameter spaces.

Moreover, in a substantial fraction of successful episodes, valid kinetic models were discovered within the first 10-20 steps of a 50-step rollout (Figure 2a). This indicates that shorter episodes may suffice in practice and highlights PPOs ability to quickly steer the search toward biologically plausible regions from random initializations.

However, the reward trajectories reveal important limitations. The highest reward within an episode often occurs well before the final timestep, suggesting that the agent either overshoots optimal regions or drifts away from them after identifying high-quality solutions. This behavior highlights opportunities for improved algorithmic design, such as early stopping criteria or mid-trajectory solution caching, to preserve and exploit promising intermediate states.

Despite its advantages, PPO remains sensitive to initialization and hyperparameter choices. As with NES, some episodes fail to converge to valid solutions. This instability suggests a need for more robust policy optimization methods or regularization techniques tailored to the structure of the kinetic modeling task.

Finally, analysis of eigenvalue distributions over training episodes (Figure 2d) reveals a progressive leftward shift in dominant eigenvalues, indicating that the agent is learning to generate increasingly stable models. This trend aligns with rising average rewards and supports the conclusion that PPO can effectively leverage feedback to refine kinetic parameter sets over time.

4.3 Ablations

The ablation study highlights the sensitivity of RL-Renaissance to training configuration. Reducing the number of training episodes (to 200 or 50) substantially degrades performance, lowering incidence rates to 0.59 and 0.52, respectively. Modifications such as cosine learning rate scheduling, altered clipping thresholds, or fewer training epochs further reduce success rates and result in unstable system dynamics, as evidenced by higher $\lambda_{\max}$ values and increased variance. The baseline configuration (100 episodes, default hyperparameters) yields the best overall trade-off between performance and stability, achieving a 0.91 incidence rate and low median eigenvalues indicative of dynamical stability.

5 Conclusion

We introduced RL-RENAISSANCE, a reinforcement learning-based alternative to the NES-based RENAISSANCE framework for kinetic model generation. By formulating parameter refinement as a sequential decision process and applying Proximal Policy Optimization (PPO), our method achieves a comparable incidence rate while requiring significantly fewer samples, demonstrating the sample efficiency and adaptability of policy-gradient methods for exploring high-dimensional biochemical parameter spaces.

Our findings show that PPO is capable of rapidly converging to biologically valid solutions from random initializations, often within a small number of refinement steps. Nevertheless, PPO introduces new challenges: its sensitivity to hyperparameter settings and a tendency to overshoot optimal regions call for improved algorithmic strategies. Future work may benefit from incorporating early stopping mechanisms, more robust reward shaping, or alternative policy optimization techniques to enhance stability and performance.

By releasing our codebase and trained models, we aim to support further research on the use of reinforcement learning in systems biology and promote more scalable, data-efficient approaches to kinetic modeling.

References

[1] S. Andreozzi, L. Miskovic, and V. Hatzimanikatis. ischrunk—in silico approach to characterization and reduction of uncertainty in the kinetic models of genome-scale metabolic networks. *Metabolic Engineering*, 33:158–168, 2016.

[2] S. Andreozzi, L. Miskovic, and V. Hatzimanikatis. ischrunk in silico approach to characterization and reduction of uncertainty in the kinetic models of genome-scale metabolic networks. *Metabolic Engineering*, 33:158–168, 2016.

[3] S. Choudhury, M. Moret, P. Salvy, D. Weilandt, V. Hatzimanikatis, and L. Miskovic. Reconstructing kinetic models for dynamical studies of metabolism using generative adversarial networks. *Nature Machine Intelligence*, 4(8):710–719, 2022.

[4] S. Choudhury, B. Narayanan, M. Moret, V. Hatzimanikatis, and L. Miskovic. Generative machine learning produces kinetic models that accurately characterize intracellular metabolic states. *bioRxiv*, 2023.

[5] S. Gopalakrishnan, S. Dash, and C. Maranas. K-fit: An accelerated kinetic parameterization algorithm using steady-state fluxomic data. *Metabolic Engineering*, 61:197–205, 2020.

[6] L. Miskovic, J. Béal, M. Moret, and V. Hatzimanikatis. Uncertainty reduction in biochemical kinetic models: Enforcing desired model properties. *PLoS computational biology*, 15(8):e1007242, 2019.

[7] L. Miskovic and V. Hatzimanikatis. Production of biofuels and biochemicals: in need of an oracle. *Trends in Biotechnology*, 28(7):391–397, 2010.

A Appendix / supplemental material