Generating accurate kinetic models via PPO refinement steps

Team Luka Doncic

M. Vukasinovic, N. Jovanovic, S. Nikolic, A. Dávid, M. Vanoušek, L. Cizinsky

supervised by Ljubisa Miskovic and Ilias Toumpe

EPFL

Context and Motivation

Kinetic models are essential for understanding how **cells** behave dynamically:

- How they respond to stress, grow, or adapt to changes?
- What happens when genes or enzymes are altered (e.g., in synthetic biology or disease)?

Applications include robustness analysis of metabolic phenotypes or bioreactor simulations (Figure 1).

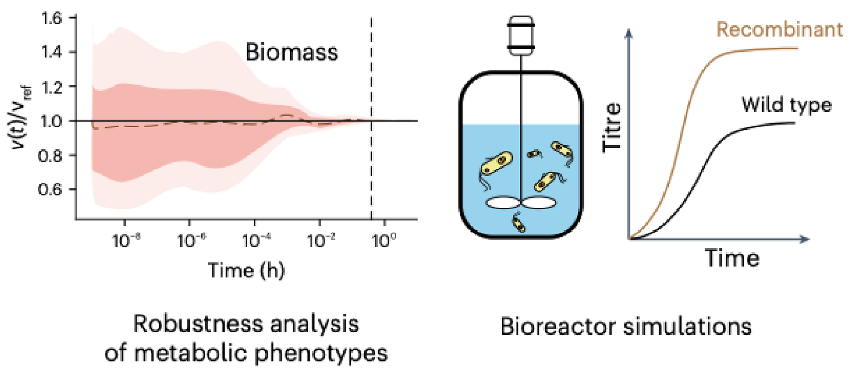

Figure 1. Showcase of usecases of kinematic models. Figure from [3].

However, these models require **thousands** of kinetic parameters, and most of them are **unknown**, **hard** to measure, or **inconsistent** across studies.

Previous Work

Traditional methods:

- use outdated parameters, rely on brute-force optimization, and assume simplified networks,
- struggle with scalability, data uncertainty, and handling multi-scale constraints.

Although attempts have been made to address these issues, they often require:

- **extensive** computational time [4],
- **ground truth training data** derived from traditional kinetic modeling approaches [1, 5, 2].

Finally, newly proposed framework **RENAISSANCE** [3] addresses both of these issues, however, it still has the following limitations:

- **unstable convergence** towards finding the optimal kinetic parameters,
- requires **per environment re-training**.

An overview of the Renaissance framework is shown in Figure 2.

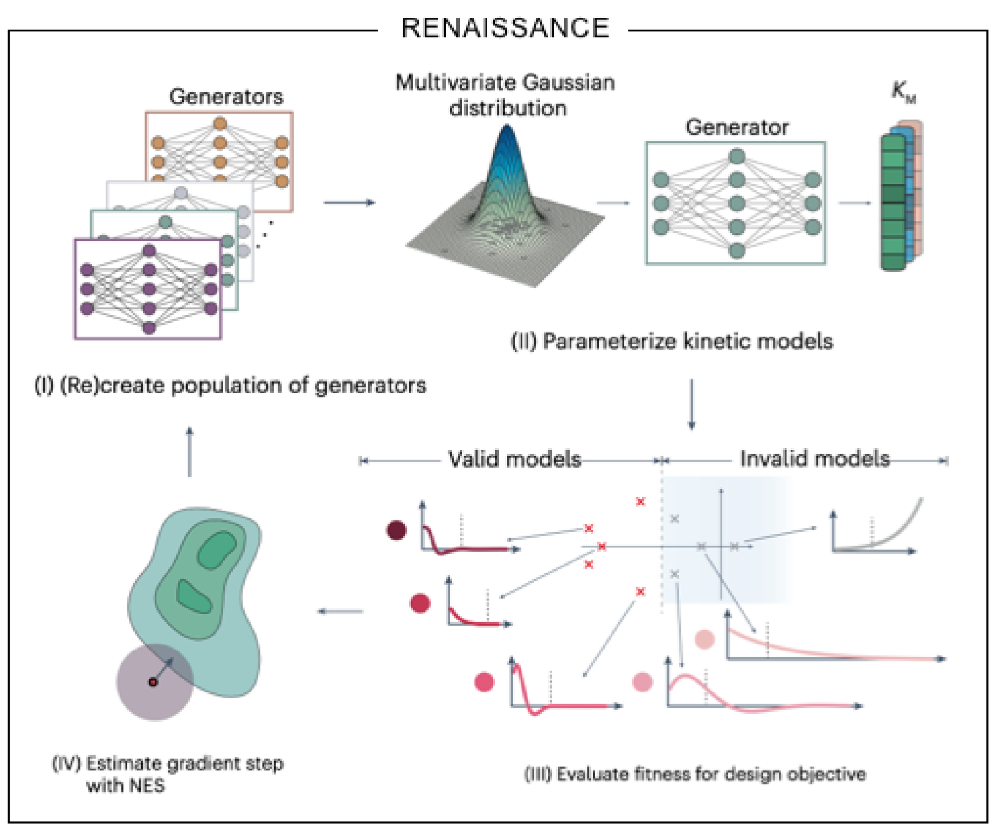

Figure 2. Overview of the Renaissance framework. Figure from [3].

Key Contributions

Our contributions can be summarized as follows:

- we propose RL based alternative to the evolutionary algorithm used in RENAISSANCE,
- we provide a comparative study of the proposed RL approach vs. the traditional evolutionary algorithm.

Method Overview

To address the limitations of RENAISSANCE, we reframe kinetic parameter generation as a **Multi-step refinement** process using Proximal Policy Optimization (PPO). The process **iteratively** refines a 384-dimensional parameter set over T steps, guided by stability-based rewards. The agent starts with an initial parameter set p_0, forms the initial state $s_0 = (p_0, 0)$, and iteratively samples actions a_t, computes rewards r_t, and transitions to the next state s_{t+1}. This process generates the full trajectory τ. Below, we outline the key components:

State (s_t): At step t, the state comprises:

- The current kinetic parameters p_t, a 384-dimensional vector.
- A timestep embedding $t \in \{0, \dots, T-1\}$.

Policy ($\pi_\theta(a_t|s_t)$): π_θ takes s_t as input and outputs a Gaussian distribution over actions:

- **Architecture**: A shared base network with two heads: one for the mean $\mu(s_t, \theta)$, another for the log standard deviation $\log \sigma$.
- **Action Sampling**: Actions are sampled as $a_t \sim \mathcal{N}(\mu(s_t, \theta), \sigma^2 I)$, where σ controls exploration.

Actions (a_t): Actions are parameter updates, applied as:

$$p_{t+1} = p_t + a_t \beta,$$

wher β is the action scaling hyper-parameter. Actions are continuous, with p_{t+1} constrained to valid ranges.

Reward (r_t): The reward guides refinement based on system stability:

$$z = \text{clip}(\lambda_{\text{max}} - \lambda_{\text{part}}, -20, +20)$$

$$r = \frac{1}{1 + e^z}$$

where $\lambda_{\text{max}}(p_t)$ is the maximum Jacobian eigenvalue and λ_{part} is the threshold for valid eigenvalues.

PPO Algorithm

The PPO algorithm trains an agent to refine parameters iteratively using an actor-critic framework:

- **Agent (Actor)**: A neural network outputs action distributions $\pi(a_t|s_t, \theta)$, guiding parameter updates to maximize rewards.
- **Critic**: A separate network estimates the value function $V(s_t)$, predicting expected cumulative rewards from state s_t.
- **Training Loop**:
  - *Trajectory Sampling*: Generating the full trajectory τ.
  - *Returns and Advantages*: Compute returns $G_t = r_t + \gamma r_{t+1} + \cdots$ with discount factor $\gamma = 0.99$. Calculate advantages $A_t = G_t - V(s_t)$ to estimate policy improvement.
  - *Policy and Critic Updates*: Optimize the clipped PPO objective and update the critic by minimizing the value loss:

This iterative training leverages PPO's stability to optimize parameter refinements, enhancing convergence and model validity.

Results

Figure 3. Mean and maximum value of the reward over the episodes

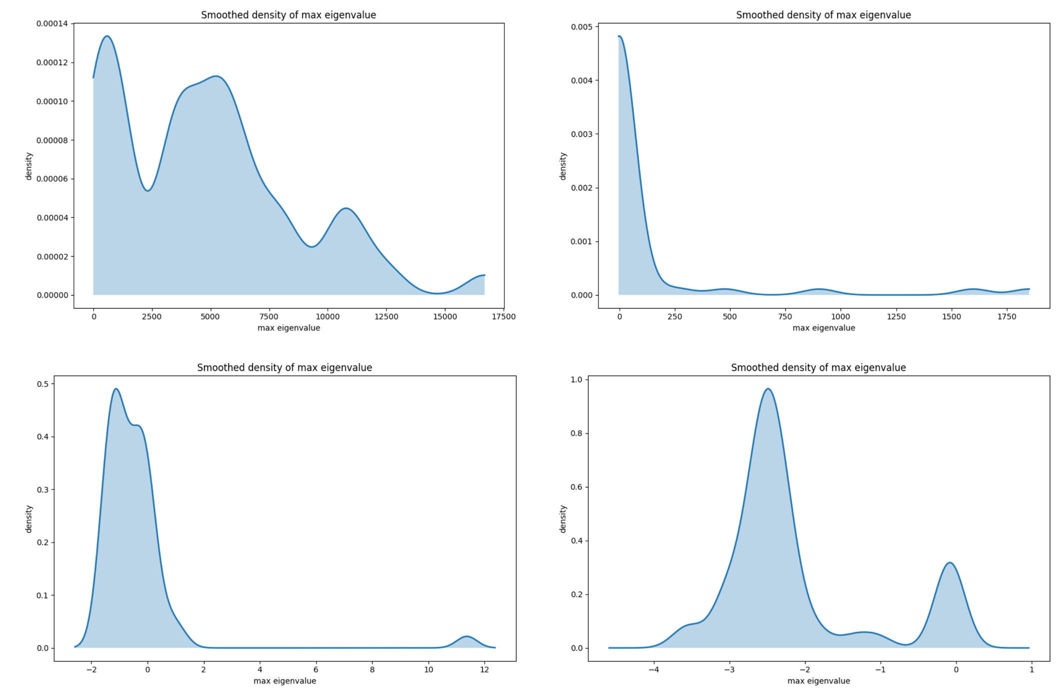

Figure 4. Distribution of eigenvalues in different episodes. Latter episodes have lower maximum eigenvalues. If the maximum eigenvalue is below 0, the solution is valid.

α_a	α_c	w_v	ϵ	g_{norm}	epochs	sched.	β	$\lambda_{\max} \downarrow$	$\mathbb{R}_{\max} \uparrow$	$\mathbb{R}_{\text{mean}} \uparrow$
3e-4	1e-3	5e-1	2e-1	5e-1	10	constant	1.0	59.14	0.54	0.17
1e-5	1e-3	5e-1	2e-1	5e-1	10	constant	1.0	66.33	0.32	0.13
1e-4	1e-3	5e-1	2e-1	5e-1	10	constant	1.0	25.58	0.24	0.13
3e-4	**1e-4**	5e-1	2e-1	5e-1	10	constant	1.0	31.52	0.37	0.13
3e-4	**5e-4**	5e-1	2e-1	5e-1	10	constant	1.0	**21.49**	0.34	0.17
3e-4	1e-3	**7.5e-1**	2e-1	5e-1	10	constant	1.0	91.00	0.30	0.10
3e-4	1e-3	**2.5e-1**	2e-1	5e-1	10	constant	1.0	45.41	**0.59**	0.17
3e-4	1e-3	5e-1	**1e-1**	5e-1	10	constant	1.0	203.05	0.29	0.09
3e-4	1e-3	5e-1	2e-1	**7.5e-1**	10	constant	1.0	36.19	0.52	**0.18**
3e-4	1e-3	5e-1	2e-1	**2.5e-1**	10	constant	1.0	27.20	0.42	0.16
3e-4	1e-3	5e-1	2e-1	5e-1	**5**	constant	1.0	32.33	0.32	0.17
3e-4	1e-3	5e-1	2e-1	5e-1	10	**cosine**	1.0	29.99	0.30	0.14
3e-4	1e-3	5e-1	2e-1	5e-1	10	constant	**0.25**	84.13	0.25	0.13

Table 1. **Ablation study results.** We have ablated learning rates of the actor (α_a) and critic (α_c), weight of the value loss (w_v), surrogate clipping (ϵ), gradient norm (g_{norm}), number of epochs per episode, learning rate scheduler and action scaler β. We report mean over the last 10 episodes of the maximum eigenvalue ($\lambda_{\max}$), maximum ($\mathbb{R}_{\max}$) and mean ($\mathbb{R}_{\text{mean}}$) reward. We use sigmoid reward function, therefore $\mathbb{R} \in (0, 1)$.

Discussion

- Our results in controlled environments confirm that PPO is a viable framework for parameter estimation in dynamic systems with stability constraints. Stability of the PPO training remained a key issue.
- For future work we will test how different reward shaping techniques and multi-trajectory approach will influence the stability of the algorithm, hopefully reducing variance.

References

- Stefano Andreozzi, Ljubisa Miskovic, and Vassily 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.
- Subham Choudhury, Michael Moret, Pierre Salvy, Daniel Weilandt, Vassily Hatzimanikatis, and Ljubisa Miskovic. Reconstructing kinetic models for dynamical studies of metabolism using generative adversarial networks. *Nature Machine Intelligence*, 4(8):710–719, 2022.
- Subham Choudhury, Bharath Narayanan, Michael Moret, Vassily Hatzimanikatis, and Ljubisa Miskovic. Generative machine learning produces kinetic models that accurately characterize intracellular metabolic states. *bioRxiv*, 2023.
- Saratram Gopalakrishnan, Satyakam Dash, and Costas Maranas. K-fit: An accelerated kinetic parameterization algorithm using steady-state fluxomic data. *Metabolic Engineering*, 61:197–205, 2020.
- Ljubisa Miskovic, Jonas Béal, Michael Moret, and Vassily Hatzimanikatis. Uncertainty reduction in biochemical kinetic models: Enforcing desired model properties. *PLoS computational biology*, 15(8):e1007242, 2019.