Enhancing reinforcement learning for population setpoint tracking in co-cultures

Sebastián Espinel-Ríos [*,†] **Joyce Qiaoxi Mo** [**]
Dongda Zhang [***] **Ehecatl Antonio del Rio-Chanona** [****]
José L. Avalos [*]

[*] *Princeton University, United States*
[**] *Princeton Satellite Systems, United States*
[***] *University of Manchester, United Kingdom*
[****] *Imperial College London, United Kingdom*
[†] *Current affiliation: Commonwealth Scientific and Industrial Research Organisation, Australia*

Abstract: Efficient multiple setpoint tracking can enable advanced biotechnological applications, such as maintaining desired population levels in co-cultures for optimal metabolic division of labor. In this study, we employ reinforcement learning as a control method for population setpoint tracking in co-cultures, focusing on policy-gradient techniques where the control policy is parameterized by neural networks. However, achieving accurate tracking across multiple setpoints is a significant challenge in reinforcement learning, as the agent must effectively balance the contributions of various setpoints to maximize the expected system performance. Traditional return functions, such as those based on a quadratic cost, often yield suboptimal performance due to their inability to efficiently guide the agent toward the simultaneous satisfaction of all setpoints. To overcome this, we propose a novel return function that rewards the simultaneous satisfaction of multiple setpoints and diminishes overall reward gains otherwise, accounting for both stage and terminal system performance. This return function includes parameters to fine-tune the desired smoothness and steepness of the learning process. We demonstrate our approach considering an *Escherichia coli* co-culture in a chemostat with optogenetic control over amino acid synthesis pathways, leveraging auxotrophies to modulate growth.

Keywords: Reinforcement learning, policy gradient, return function, setpoint tracking, co-cultures, optogenetics.

1. INTRODUCTION

The genetic engineering of microorganisms has enabled the manufacturing of a wide range of products, including chemicals, fuels, materials, and pharmaceuticals (Luo et al., 2021). Traditionally, bioproduction has relied on monocultures, harnessing the metabolic capabilities of a single species. However, engineering (large) metabolic pathways within a single cell can lead to *metabolic burden*, compromising cell growth and the process volumetric productivity. To alleviate this burden and enhance the overall process efficiency, metabolic pathways can be partitioned among multiple microbial species or engineered strains in a consortium (*division of labor*) (Roell et al., 2019). By distributing metabolic submodules across different populations, each member can specialize in the part of the pathway it is best suited for. Maintaining specific population levels within the consortium is essential for optimizing the production process, as the concentration of each specialized cell directly influences the achievable volumetric productivity of the metabolic submodule it carries.

Controlling population levels in microbial consortia, however, poses significant challenges. Due to the *competitive exclusion principle* (Kneitel, 2019), when multiple microorganisms compete for a single limiting resource, e.g., the carbon source, the consortium member with the highest growth rate (*fitness*) will eventually outcompete and displace the others. Adjusting initial inoculation ratios can *alleviate* competitive exclusion in the *short term*, but it does not alter the long-term dynamics, making this approach unsuitable for prolonged cultivations, such as those in continuous bioreactors. Engineering endogenous interactions, like mutualism, is another strategy to address competitive exclusion by creating co-dependency among cells, e.g., via engineering auxotrophies and cross-feeding relationships (Peng et al., 2024). However, the population dynamics are predetermined by the engineered interactions, limiting operational flexibility and adaptability.

To overcome these limitations, external control mechanisms have been proposed to enhance operational flexibility and facilitate feedback control via externally tunable inputs. This approach allows the user to define various population setpoints (i.e., constant reference values) according to process requirements, rather than relying on *predefined engineered* setpoints as in endogenous interactions. Conventional control strategies, such

[⋆] This work was supported by the US DOE-BER (DE-SC0022155) and US NSF (MCB-2300239).
[⋆⋆] SER e-mail: sebastian.espinelrios@csiro.au.

as proportional-integral-derivative (PID) controllers, have been proposed to regulate population ratios in bioreactors (Gutiérrez Mena et al., 2022). Although simple and useful in certain cases, PID controllers are inherently *reactive*, struggle with nonlinear system dynamics, and cannot handle system constraints, motivating more advanced control approaches. In the context of microbial consortia, model predictive control (MPC) offers a more advanced alternative by utilizing a system model to compute control actions that minimize a cost function (Espinel-Ríos et al., 2023). However, its implementation may be challenging if obtaining accurate mathematical models is difficult or when dealing with complex models (e.g., stochastic, highly nonlinear, stiff, or discontinuous dynamics).

Reinforcement learning (RL) is a promising machine-learning control strategy where an agent (the *controller*) learns optimal control actions (*process inputs*) through interactions with the environment (the *bioreactor system*). Previous studies have considered Q-learning, an action-value method, for setpoint tracking in microbial consortia with discrete bang-bang feeding control actions (Treloar et al., 2020). However, Q-learning involves *deterministic* policies and requires careful balancing of exploration and exploitation (often *hard-coded* via epsilon-greedy strategies). The *value function* in Q-learning, the expected cumulative reward, from which the optimal actions are computed upon solving an optimization problem, can be approximated using, e.g., neural networks. Q-learning may struggle to converge to an optimal policy if the value function is not properly approximated and/or if the optimization step poses numerical difficulties. This can be particularly challenging in continuous or high-dimensional action spaces. Thus, Q-learning may be more suitable for discrete actions.

In this work, we consider RL based on the policy-gradient method (Petsagkourakis et al., 2020), which can address several of the limitations of Q-learning. Policy-gradient methods directly optimize the *control policy*, which can be approximated using, e.g., neural networks. This direct approach focused on the policy itself ensures convergence to at least a local optimum and naturally accommodates continuous input variables, enhancing operational flexibility as more of the input space can be explored and exploited. Furthermore, policy-gradient methods involve *stochastic* policies, which naturally balance the agent's adaptive exploration-exploitation over time and are better suited for handling systems with high stochasticity, such as biological processes. Even when dealing with deterministic systems, a stochastic-by-design policy can favor exploration and help to escape local minima. Overall, RL offers a promising approach for controlling complex systems that are difficult to differentiate using conventional model-based optimization methods, while enabling the development of uncertainty-aware policies.

A critical component of RL is the design of the *return function*, which guides the agent toward desired optimal behaviors. The inverse quadratic cost function, commonly used in optimal control problems, may offer good convergence properties when tracking a *single* setpoint in RL. However, in the context of microbial consortia, the latter function lacks a mechanism to directly incentivize the simultaneous satisfaction of multiple independent setpoint objectives [1]. In other words, the agent may need to explore more extensively to discover a *sweet-spot* scenario where all setpoints are satisfied without prioritizing one over the other. This often results in a higher risk of suboptimal performance as the agent might oscillate between multiple *individual* objectives.

To tackle this issue, we propose a novel return function based on multiplicative inverse *saturation* functions that can enhance multiple setpoint tracking performance. This reward structure ensures that maximum reward is achieved only when all setpoints are satisfied simultaneously *and* that improving individual setpoints while others remain off-target *diminishes* overall reward gains. This promotes a more balanced progression toward multiple targets and can guide the agent more precisely. The return function can be shaped to balance *smoothness* and *steepness* in the policy gradients and in the updates of its parameters.

2. CO-CULTURE CASE STUDY WITH OPTOGENETIC CONTROL OF GROWTH

As a case study, we consider a two-member *Escherichia coli* consortium, where each microorganism has an auxotrophy for a specific amino acid (Fig. 1). One strain, *E. coli* 1, is auxotrophic for lysine due to the deletion of *lysA* (diaminopimelate decarboxylase). The other strain, *E. coli* 2, is auxotrophic for leucine due to the deletion of *leuA* (2-isopropylmalate synthase). It is assumed that *lysA* can be optogenetically induced with blue light in *E. coli* 1, leveraging the PBLind-v1 system (Jayaraman et al., 2016). Similarly, *leuA* can be optogenetically induced with red light in *E. coli* 2 using the pREDawn-DsRed system (Multamäki et al., 2022). For simplicity, we assume that the rate of auxotrophic amino acid synthesis can be directly linked to the light inputs, lumping the dynamics of enzyme expression (*lysA* and *leuA*). Additionally, we consider that amino acid synthesis does not lead to excretion, as amino acids accumulate only to normal physiological levels. The system dynamics thus follow:

$$\frac{\mathrm{d}s}{\mathrm{d}t} = -q_{s_1}b_1 - q_{s_2}b_2 + (s_{\mathrm{in}} - s)d_l, \tag{1a}$$

$$\frac{\mathrm{d}b_i}{\mathrm{d}t} = (\mu_i - d_l)b_i, \quad \forall i \in \{1,2\}, \tag{1b}$$

$$\frac{\mathrm{d}a_i}{\mathrm{d}t} = q_{a_i} - (d_{a_i} + \mu_i)a_i, \quad \forall i \in \{1,2\}, \tag{1c}$$

with growth, substrate uptake, and lumped transcription/translation kinetic rate functions:

$$\mu_i = \mu_{\max_i}\left(\frac{s}{s+k_{s_i}}\right)\left(\frac{f_c a_i}{f_c a_i + k_{a_i}}\right), \quad \forall i \in \{1,2\}, \tag{2a}$$

$$q_{s,i} = Y_{s/b_i}\mu_i, \quad \forall i \in \{1,2\}, \tag{2b}$$

$$q_{a,i} = q_{a_{\max_i}}\left(\frac{I_i^{n_i}}{I_i^{n_i} + k_{I_i}^{n_i}}\right), \quad \forall i \in \{1,2\}, \tag{2c}$$

where the concentration of *E. coli* i in g/L, the intracellular concentration of the amino acid counteracting the auxotrophy in species i in mmol/g, and the concentration of the shared carbon source (glucose) in mmol/L are denoted by $b_i \in \mathbb{R}$, $a_i \in \mathbb{R}$, and $s \in \mathbb{R}$, respectively. The control inputs are the blue and red light intensities, denoted by $I_1 \in \mathbb{R}$ and $I_2 \in \mathbb{R}$, respectively. Here, $\mu_{\max_i}$, k_{s_i}, f_c,

[1] By multiple *setpoint* tracking, we refer to tracking *constant* reference values for different state variables.

k_{a_i}, Y_{s/b_i}, $q_{a_{\max_i}}$, n_i, k_{I_i}, and d_{a_i} are constant parameter values, d_l is the dilution rate of the chemostat, and s_{in} is the substrate concentration at the inflow.

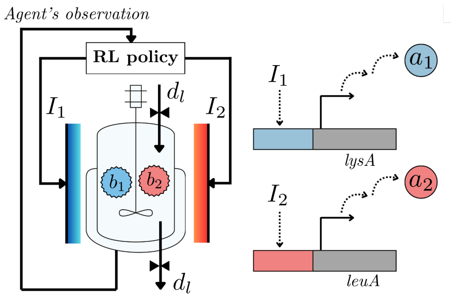

Fig. 1. *E. coli* co-culture in a chemostat with optogenetic control of *lysA* and *leuA* synthesis.

3. REINFORCEMENT LEARNING VIA POLICY GRADIENTS

Let us denote the dynamic states of the system by $\boldsymbol{x} \in \mathbb{R}^{n_x}$ and the system inputs by $\boldsymbol{u} \in \mathbb{R}^{n_u}$. The dynamic behavior of the system is formulated as a Markov decision process, where the transition from time t to $t+1$ is described by a *probability distribution* conditioned on the system state $\boldsymbol{x}_t$ and input $\boldsymbol{u}_t$. This transition can be approximated by the function $f_x : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$, incorporating stochasticity through *random noise* $\boldsymbol{d}_t \in \mathbb{R}^{n_x}$. Therefore, the system dynamics are given by:

$$\boldsymbol{x}_{t+1} = f_x(\boldsymbol{x}_t, \boldsymbol{u}_t) + \boldsymbol{d}_t. \tag{3}$$

Note that the state vector $\boldsymbol{x}$ captures, in principle, all the information necessary to predict the dynamics of the system. Time is discretized in equidistant intervals, $t \in [t_0, t_1, ..., t_f]$.

The *probability distribution* of the input actions is defined by the control policy π, parameterized by $\boldsymbol{\theta} \in \mathbb{R}^{n_\theta}$. Thus, the control action at time t is sampled from:

$$\boldsymbol{u}_t \sim \pi(\boldsymbol{u}_t \mid \boldsymbol{s}_t, \boldsymbol{\theta}), \tag{4}$$

where $\boldsymbol{s}_t \in \mathbb{R}^{n_s}$ represents the *agent's observation* of the system, thus *measurable*. Note that $\boldsymbol{s}_t$ is a more abstract variable that may not necessarily be the full *dynamic system state* $\boldsymbol{x}_t$. That is, the agent may incorporate additional or different information to better understand the system when making decisions, such as past *measured* state/input pairs or the process sampling time.

We represent with $\boldsymbol{\tau} \in \mathbb{R}^{n_\tau}$ the joint trajectory of *observed* states, actions, and rewards $R \in \mathbb{R}$ over the time course of the process, i.e., $\boldsymbol{\tau} = \{(\boldsymbol{s}_t, \boldsymbol{u}_t, R_{t+1}, \boldsymbol{s}_{t+1})\}_{t=0}^{t_f-1}$, with probability distribution:

$$\mathrm{P}(\boldsymbol{\tau} \mid \boldsymbol{\theta}) = \mathrm{P}(\boldsymbol{s}_0) \prod_{t=0}^{t_f-1} \left[\pi(\boldsymbol{u}_t \mid \boldsymbol{s}_t, \boldsymbol{\theta}) \mathrm{P}(\boldsymbol{x}_{t+1} \mid \boldsymbol{x}_t, \boldsymbol{u}_t)\right]. \tag{5}$$

We denote the agent's *return function* or *performance metric* with the function $J : \mathbb{R}^{n_\tau} \to \mathbb{R}$, which accounts for the rewards over $\boldsymbol{\tau}$. In RL, the agent aims to maximize the *expected return* by finding an optimal policy:

$$\max_{\pi(\cdot)} \mathbb{E}_{\boldsymbol{\tau}}\left[J(\boldsymbol{\tau})\right], \tag{6}$$

where $\mathbb{E}_{\boldsymbol{\tau}}$ represents the expected value over $\boldsymbol{\tau}$ given the policy $\pi(\cdot)$.

We parameterize the mean $\boldsymbol{m}_t \in \mathbb{R}^{n_u}$ and standard deviation $\boldsymbol{\sigma}_t \in \mathbb{R}^{n_u}$ of the policy using deep neural networks $f_{\text{DNN}} : \mathbb{R}^{n_s} \times \mathbb{R}^{n_\Theta} \to \mathbb{R}^{n_u} \times \mathbb{R}^{n_u}$ with parameters $\Theta \in \mathbb{R}^{n_\Theta}$:

$$\boldsymbol{m}_t, \boldsymbol{\sigma}_t = f_{\text{DNN}}(\boldsymbol{s}_t, \boldsymbol{\Theta}). \tag{7}$$

Thus, $\boldsymbol{\theta} := \boldsymbol{\Theta}$.

In policy-gradient RL, the parameters of the policy are updated following the gradient ascent:

$$\boldsymbol{\theta}_{m+1} = \boldsymbol{\theta}_m + \alpha \nabla_{\boldsymbol{\theta}} \mathbb{E}_\tau\left[J(\boldsymbol{\tau})\right], \tag{8}$$

where m is an instance of parameter update, i.e., an *epoch*, over $N_{\text{epoch}} \in \mathbb{N}$ epochs, and $\alpha \in \mathbb{R}$ is the learning rate. The first update occurs at $m = 0$.

Based on the Policy Gradient Theorem (Sutton et al., 1999):

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_\tau\left[J(\boldsymbol{\tau})\right] = \int \mathrm{P}(\boldsymbol{\tau} \mid \boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \log\left(\mathrm{P}(\boldsymbol{\tau} \mid \boldsymbol{\theta})\right) J(\boldsymbol{\tau})\, \mathrm{d}\boldsymbol{\tau}, \tag{9}$$

thus,

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\tau}}\left[J(\boldsymbol{\tau})\right] = \mathbb{E}_{\boldsymbol{\tau}}[J(\boldsymbol{\tau}) \nabla_{\boldsymbol{\theta}} \log\left(\mathrm{P}(\boldsymbol{\tau} \mid \boldsymbol{\theta})\right)]. \tag{10}$$

Combining Eqs. (5) and (10) leads to:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\tau}}\left[J(\boldsymbol{\tau})\right] = \mathbb{E}_{\boldsymbol{\tau}}\left[J(\boldsymbol{\tau}) \nabla_{\boldsymbol{\theta}} \sum_{t=0}^{t_f-1} \log\left(\pi\left(\boldsymbol{u}_t \mid \boldsymbol{s}_t, \boldsymbol{\theta}\right)\right)\right]. \tag{11}$$

In this work, we approximate the expectation in Eq. (11) through Monte Carlo simulations:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\tau}}\left[J\left(\boldsymbol{\tau}\right)\right] \approx \frac{1}{N_{\text{MC}}} \sum_{k=1}^{N_{\text{MC}}} \left[\frac{J\left(\boldsymbol{\tau}^{(k)}\right) - \bar{J}\left(\boldsymbol{\tau}\right)}{\sigma_J + \varepsilon} \times \nabla_{\boldsymbol{\theta}} \sum_{t=0}^{t_f-1} \log\left(\pi\left(\boldsymbol{u}_t^{(k)} \mid \boldsymbol{s}_t^{(k)}, \boldsymbol{\theta}\right)\right)\right], \tag{12}$$

where the *episode* k in $N_{\text{MC}} \in \mathbb{N}$ Monte Carlo episodes is denoted by the superscript (k). Note that we normalize the return by subtracting the mean return $\bar{J}\left(\boldsymbol{\tau}\right)$ and dividing by the standard deviation of the return σ_J across episodes, with a small constant ε added for numerical stability. It is worth noting that the gradient of the log-probability informs how changing the parameters of the policy affects the probability of taking that action. Furthermore, values of $J\left(\boldsymbol{\tau}^{(k)}\right) > \bar{J}\left(\boldsymbol{\tau}\right)$ will encourage updating the policy's parameters in a way that increases the probability of the actions taken in that episode, thereby reinforcing better-than-average returns. Conversely, values of $J\left(\boldsymbol{\tau}^{(k)}\right) < \bar{J}\left(\boldsymbol{\tau}\right)$ will favor updating the policy's parameters to decrease the probability of those actions, thereby discouraging worse-than-average returns.

4. RETURN FUNCTION DESIGN

For setpoint tracking of the individual populations in the co-culture outlined in Section 2, we consider four possible return function designs.

- **Case 1**. This design uses the stage quadratic objective as the return function:

$$J = -\sum_{t=1}^{t_f} \left[w_1(b_{1_t} - b_1^*)^2 + w_2(b_{2_t} - b_2^*)^2\right], \tag{13}$$

where b_1^* and b_2^* are the constant setpoint references for the respective biomass populations, and w_1 and w_2

are appropriate weights. Case 1 serves as a *benchmark* return function. Eq. (13) in Eq. (6) involves a *weighted multi-objective optimization*, and there is no direct mechanism to favor the simultaneous satisfaction of both setpoints. This means that one can accumulate rewards even if only one setpoint improves, while the other does not, which can lead to stagnant learning. In that sense, if the weights are not *tuned* properly, the agent may be biased or misled towards optimizing only one setpoint. Note that Eq. (13) deals with the *inverse* quadratic cost, hence the negative sign.

- **Case 2**. This design introduces multiplicative inverse *saturation* functions in the return. The overall return is divided into a *stage* reward accumulated from $t = 1$ until $t = t_f - 1$ and a *terminal* or *arrival* reward at the final time step $t = t_f$:
$$J = \sum_{t=1}^{t_f-1} [w_t q_{V_t}] + w_{t_f} q_{V_{t_f}}, \tag{14}$$
where for a given time $t \in \{t_1, ..., t_f\}$:
$$q_{V_t} = \beta_{V_{\max}} \cdot \frac{\beta_{e_1}}{\beta_{e_1} + e_{1_t}} \cdot \frac{\beta_{e_2}}{\beta_{e_2} + e_{2_t}}, \tag{15}$$
and
$$e_{i_t} = (b_{i_t} - b_i^*)^2, \quad \forall i \in \{1, 2\}. \tag{16}$$
The maximum return at a given sampling time is determined by a tunable parameter $\beta_{V_{\max}}$. The *inverse saturation* functions lead to a decrease in reward as the quadratic error of the biomass populations increases. The steepness of these functions is controlled by the tunable parameters β_{e_1} and β_{e_2}, providing flexibility in how sharply the return improves with decreasing error. When the error e_{i_t} approaches zero, the inverse saturation function associated with population i approaches one and the maximum reward is achieved, i.e., $q_{V_{i_t}} = \beta_{V_{\max}}$. This setup ensures that simultaneous setpoint satisfaction yields the highest reward.
Let us consider an extreme scenario, assuming $\beta_{e_1} = \beta_{e_2}$ for simplicity, where one biomass population is exactly at the target but the other one remains significantly off the target, then the overall stage reward gain will approach zero at that point. The incorporation of both *stage* and *terminal* rewards helps to balance transient and final multi-setpoint tracking performance, determined by the weights w_t, $\forall t \in \{1, ..., t_f\}$.
In the context of the co-culture case study, we select $\beta_{e_1} = \beta_{e_2} = 3$ in Eq. (15) for **Case 2**.
- **Case 3**. This design follows the same approach as Case 2, but with $\beta_{e_1} = \beta_{e_2} = 9$ in Eq. (15).
- **Case 4**. This design also follows the approach of Case 2, but with $\beta_{e_1} = \beta_{e_2} = 27$ in Eq. (15).

The key individual functions contributing to the returns for Cases 1-4 are shown in Fig. 2. As seen in the figure, the functions exhibit varying degrees of steepness and smoothness depending on the distance to the target.

5. SETPOINT TRACKING RESULTS IN THE CO-CULTURE

We compare the performance of the return functions described in the previous section for multi-setpoint tracking

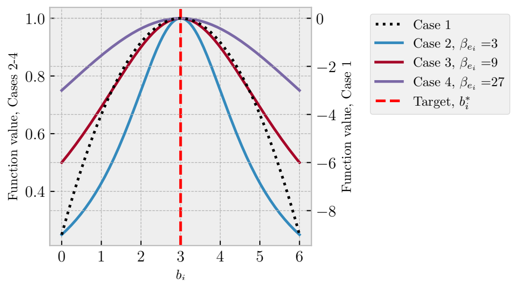

Fig. 2. Comparison of key individual elements contributing to the return functions in Cases 1-4. For Case 1, the inverse quadratic error term $-(b_{i_t} - b_i^*)^2$ is plotted, representing the penalty for deviations from the target. For Cases 2-4, the inverse saturation function $\frac{\beta_{e_i}}{\beta_{e_i}+e_{i_t}}$ is shown. The target for population i is $b_i^* = 3\,\mathrm{g/L}$.

of populations in our case study using RL. The setpoints are $b_1^* = b_2^* = 3\,\mathrm{g/L}$. The learning process is conducted in PyTorch (Paszke et al., 2019) over 350 epochs, 500 Monte Carlo episodes each, with a constant learning rate of 0.001. The policy is parameterized using a deep neural network consisting of 4 hidden layers with 20 nodes each, employing the LeakyReLU activation function. There are two output linear layers: one producing the means and the other producing the standard deviations of the inputs. The dynamic model in Eqs. (1a)-(2c) is used to simulate the chemostat bioreactor in CasADi (Andersson et al., 2019). We assume full state observability and use two past state/input pairs and a time embedding normalized in the range $[-1, 1]$ as the agent's observations, resulting in the vector $s_t := [\boldsymbol{x}_{t-1}^\top, \boldsymbol{u}_{t-2}^\top, \boldsymbol{x}_t^\top, \boldsymbol{u}_{t-1}^\top, t_t^*]^\top$, where t_t^* represents the time embedding. Given our focus on return function design, we assume a deterministic plant for simplicity; yet we maintain a stochastic policy to promote *natural* exploration and avoid local minima. The simulation is carried out over 18 equally spaced time steps of 1 hour each, i.e., $t_f = 18\,\mathrm{h}$. For Cases 2-3, $\beta_{\max} = 1$, and the weights $w_1 = w_2 = ... = w_{t_f-1} = 1$. In addition, we set $w_{t_f} = 2$ to promote stability toward the endpoint of the process.

The *initial* values for these states were set to $s = 5.5\,\mathrm{mmol/L}$, $b_1 = b_2 = 0.005\,\mathrm{g/L}$, $a_1 = 1.545 \times 10^{-2}\,\mathrm{mmol/g}$, and $a_2 = 1.655 \times 10^{-3}\,\mathrm{mmol/g}$. The model parameters are $\mu_{\max_1} = \mu_{\max_2} = 0.982\,\mathrm{h}^{-1}$, $k_{s_1} = k_{s_2} = 2.964 \times 10^{-4}\,\mathrm{mmol/L}$, $f_c = 1100\,\mathrm{g/L}$, $k_{a_1} = 1.7\,\mathrm{mmol/L}$, $k_{a_2} = 0.182\,\mathrm{mmol/L}$, $Y_{s/b_1} = Y_{s/b_2} = 10.18\,\mathrm{mmol/g}$, $q_{a_{\max_1}} = 0.337\,\mathrm{mmol/g/h}$, $q_{a_{\max_2}} = 0.036\,\mathrm{mmol/g/h}$, $n_1 = 2$, $k_{I_1} = 1.052\,\mathrm{W/m^2}$, $n_2 = 4.865$, $k_{I_2} = 1.34\,\mu\mathrm{W/cm^2}$, $d_l = 0.15\,h^{-1}$, $s_{\text{in}} = 200\,\mathrm{mmol/L}$.

Fig. 3 shows the evolution of the return function over 350 epochs for Cases 1–4. The biomass profiles for the corresponding best-performing epochs (i.e., yielding the highest mean return values) are presented in Fig. 4. Case 1 shows a very steep improvement in the return function shortly after 100 epochs, then slows down until stagnating around 200 epochs, with slight oscillations thereafter. As expected, the setpoints were not properly reached at the *best* epoch for Case 1. The system initially overshoots the target, then undershoots it, and does not stabilize

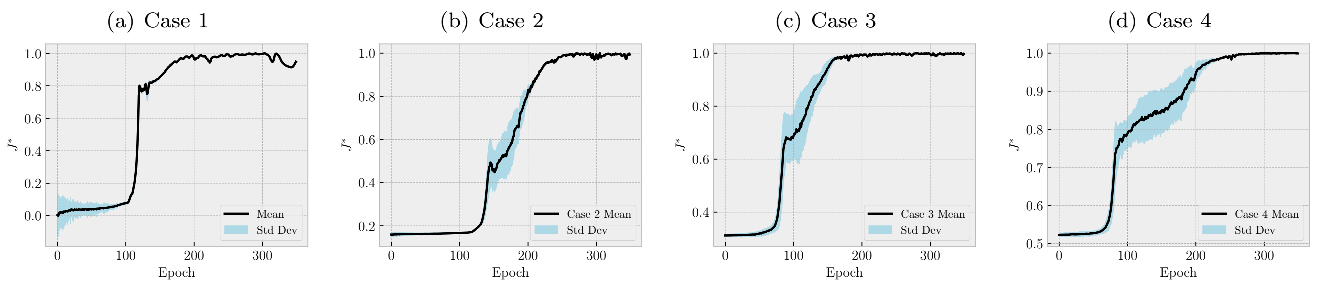

Fig. 3. Normalized return function J^* over 350 epochs for Cases 1-4. Both the mean and standard deviation (Std Dev) are shown.

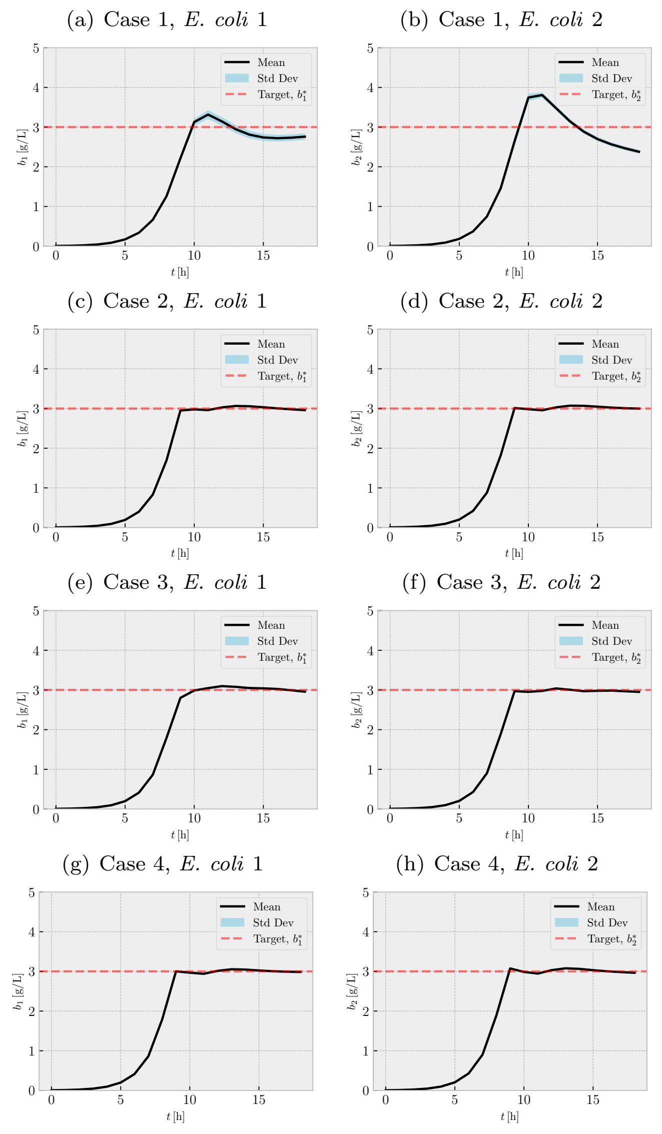

Fig. 4. Setpoint tracking performance of population levels in the co-culture for Cases 1-4. Both the mean and the standard deviation (Std Dev) are shown. These scenarios correspond to the epochs with the highest return function for each scenario (cf. Figure 3).

at the desired references over the considered epochs. In contrast, the proposed return function manages to stabilize the biomass populations at the desired setpoint references in all remaining cases, i.e., Cases 2-4. This is due to the emphasis on the simultaneous satisfaction of the setpoints, where the overall reward is diminished if any state is off-target. The main difference among these scenarios is how fast the system converges to the setpoint references. For example, Cases 2-4 plateau around 250, 180, and 250 epochs, respectively.

The high smoothness of the return function in Case 4 (cf. Fig. 2) makes the agent experience less *aggressive* parameter updates (cf. Eq. (8)), explaining its slow convergence. This scenario offers a *safe* strategy when smooth convergence and learning stability are prioritized, despite the larger number of epochs required. In contrast, Case 3 finds the *best* balance between steepness and smoothness in the return function, leading to the fastest convergence. Notably, although Case 2 has the steepest shape (cf. Fig. 2), it takes about the same number of epochs as Case 4 to stabilize (the latter having the smoothest return function shape). This can be explained by the fact that, similar to Case 1, high steepness can lead to more aggressive parameter updates and, consequently, less efficient learning.

Overall, these results underscore the importance of appropriate return function design for efficient RL, particularly in policy-gradient approaches and in multiple setpoint tracking problems. One advantage of our proposed return function design is that it can be *shaped* by the user to achieve either steeper or smoother convergence. Furthermore, this approach can unlock schemes with online adaptation of the proposed return function's parameters to better guide the agent's learning. Such an adaptive strategy would play a similar role to an *adaptive learning rate* in Eq. (8), but with the advantage of being directly tailored to the return function, making it more *interpretable*.

Finally, for demonstration purposes, we show in Fig. 5 the dynamic profiles of the inputs and intracellular amino acids for which the microorganisms are auxotrophic. As expected, amino acid accumulation rates correlate with light intensities. Also, as observed in Fig. 6, the agent successfully maintains the system at the setpoint references by regulating the growth rates to match the bioreactor's dilution rate once the desired biomass concentrations are reached. In other words, the agent learns to influence the system's transient dynamics to drive it toward the desired *steady state*, where the rate of new cell generation balances with the rate of cell removal.

It is worth noting that the presented RL strategy is, in principle, model-free. However, dynamic models, even if only approximations of the real system, can be used to pretrain the policy offline before interaction with the real system. In addition, *domain randomization* during train-

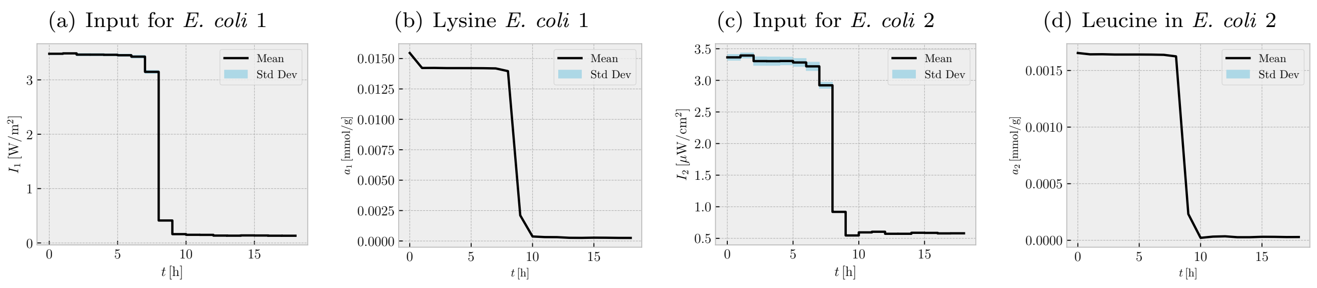

Fig. 5. Input and amino acid trajectories for the epoch with the highest mean return function value in Case 3.

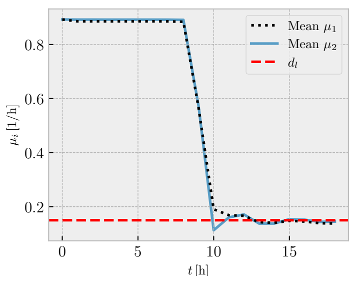

Fig. 6. Mean growth rates for the epoch with the highest mean return function value in Case 3.

ing, i.e., augmenting the system with *a priori known* or *expected* uncertainties, such as disturbances, stochastic dynamics, and variable initial conditions, can be adopted to improve the robustness of the control policy. Furthermore, offline RL strategies can be used to derive policies from *process data* without active interaction with the environment, or to perform *behavioral cloning* based on available *expert* policies.

6. CONCLUSION

In this work, we presented a novel return function design for policy-gradient RL tailored for setpoint tracking of multiple targets, such as in the case of population control in co-cultures. The proposed return function explicitly rewards the simultaneous satisfaction of multiple setpoints, leading to improved performance compared to the standard quadratic cost function, which served as a benchmark. Moreover, it can be tuned by adjusting appropriate parameters, enabling control over the smoothness and steepness of the learning process. The outlined approach can facilitate the development and application of RL for microbial consortia in biotechnological production. Future work will assess the robustness of this method under uncertainty, particularly in scenarios involving multiple setpoint and trajectory tracking (time-varying references) in microbial consortia. Another promising direction is the adaptive tuning of the return function's parameters to enhance learning efficiency and convergence.

REFERENCES

Andersson, J.A.E., Gillis, J., Horn, G., Rawlings, J.B., and Diehl, M. (2019). CasADi – A software framework for nonlinear optimization and optimal control. *Math Program Comput*, 11(1), 1–36.

Espinel-Ríos, S., Bettenbrock, K., Klamt, S., Avalos, J.L., and Findeisen, R. (2023). Machine learning-supported cybergenetic modeling, optimization and control for synthetic microbial communities. In *Computer Aided Chemical Engineering*, volume 52, 2601–2606. Elsevier.

Gutiérrez Mena, J., Kumar, S., and Khammash, M. (2022). Dynamic cybergenetic control of bacterial co-culture composition via optogenetic feedback. *Nat Commun*, 13(1), 4808.

Jayaraman, P., Devarajan, K., Chua, T.K., Zhang, H., Gunawan, E., and Poh, C.L. (2016). Blue light-mediated transcriptional activation and repression of gene expression in bacteria. *Nucleic Acids Res*, 44(14), 6994–7005.

Kneitel, J.M. (2019). Gause's competitive exclusion principle. In *Encyclopedia of Ecology*, 110–113. Elsevier.

Luo, Z.W., Ahn, J.H., Chae, T.U., Choi, S.Y., Park, S.Y., Choi, Y., Kim, J., Prabowo, C.P.S., Lee, J.A., Yang, D., Han, T., Xu, H., and Lee, S.Y. (2021). Metabolic engineering of *Escherichia coli*. In J. Nielsen, G. Stephanopoulos, and S.Y. Lee (eds.), *Metabolic Engineering*, 339–402. Wiley, 1 edition.

Multamäki, E., García De Fuentes, A., Sieryi, O., Bykov, A., Gerken, U., Ranzani, A.T., Köhler, J., Meglinski, I., Möglich, A., and Takala, H. (2022). Optogenetic control of bacterial expression by red light. *ACS Synth Biol*, 11(10), 3354–3367.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). *PyTorch: an imperative style, high-performance deep learning library.* Curran Associates Inc., Red Hook, NY, USA.

Peng, H., Darlington, A.P.S., South, E.J., Chen, H.H., Jiang, W., and Ledesma-Amaro, R. (2024). A molecular toolkit of cross-feeding strains for engineering synthetic yeast communities. *Nat Microbiol*, 9(3), 848–863.

Petsagkourakis, P., Sandoval, I., Bradford, E., Zhang, D., and Del Rio-Chanona, E. (2020). Reinforcement learning for batch bioprocess optimization. *Comput Chem Eng*, 133, 106649.

Roell, G.W., Zha, J., Carr, R.R., Koffas, M.A., Fong, S.S., and Tang, Y.J. (2019). Engineering microbial consortia by division of labor. *Microb Cell Fact*, 18(1), 35.

Sutton, R.S., McAllester, D., Singh, S., and Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. In S. Solla, T. Leen, and K. Müller (eds.), *Advances in Neural Information Processing Systems*, volume 12. MIT Press.

Treloar, N.J., Fedorec, A.J.H., Ingalls, B., and Barnes, C.P. (2020). Deep reinforcement learning for the control of microbial co-cultures in bioreactors. *PLOS Comput Biol*, 16(4), e1007783.