POLICY OPTIMIZATION IN BAYESIAN NETWORK HYBRID MODELS OF BIOMANUFACTURING PROCESSES

Hua Zheng, Wei Xie*
Department of Mechanical and Industrial Engineering
Northeastern University
Boston, MA 02115

Ilya O. Ryzhov
Robert H. Smith School of Business
University of Maryland
College Park, MD 20742

Dongming Xie
Department of Chemical Engineering
University of Massachusetts Lowell
Lowell, MA 01854

ABSTRACT

Biopharmaceutical manufacturing is a rapidly growing industry with impact in virtually all branches of medicine. Biomanufacturing processes require close monitoring and control, in the presence of complex bioprocess dynamics with many interdependent factors, as well as extremely limited data due to the high cost and long duration of experiments. We develop a novel model-based reinforcement learning framework that can achieve human-level control in low-data environments. The model uses a probabilistic knowledge graph to capture causal interdependencies between factors in the underlying stochastic decision process, leveraging information from existing kinetic models from different unit operations while incorporating real-world experimental data. We then present a computationally efficient, provably convergent stochastic gradient method for policy optimization. Validation is conducted on a realistic application with a multi-dimensional, continuous state variable.

Keywords biomanufacturing · reinforcement learning · policy optimization · Bayesian networks

1 Introduction

This work is motivated by the problem of process control in biomanufacturing, a rapidly growing industry that generated over $300 billion in revenue in 2019. Over 40% of the products in the pharmaceutical industry's development pipeline are bio-drugs (Rader 2013), designed for prevention and treatment of diseases such as cancer, Alzheimer's disease, and most recently COVID-19 (Zhou et al. 2014, Le et al. 2020). These drugs are manufactured in living cells whose biological processes are complex and highly variable. Furthermore, a typical biomanufacturing production process consists of numerous unit operations, e.g., cell culture, purification, and formulation, each of which directly impacts the outcomes of successive steps. Thus, in order to improve productivity and ensure drug quality, the process must be controlled as a whole, from end to end.

In this industry, production processes are very complex, and experimental data are very limited, because analytical testing times for complex biopharmaceuticals are very long. Unfortunately, "big data" do not exist in this industry: in a typical application, a process controller may have to make decisions based on 10 or fewer prior experiments. Not surprisingly, human error is frequent in biomanufacturing, accounting for 80% of deviations (Cintron 2015). In this paper, we propose an optimization framework, based on reinforcement learning (RL), which demonstrably improves process control in the small-data setting. Our approach incorporates domain knowledge of the underlying process mechanisms, as well as experimental data, to provide effective control policies that outperform both existing analytical methods as well as human domain experts.

*Corresponding author. Email: w.xie@northeastern.edu

As discussed in an overview by Hong et al. (2018), biomanufacturing has traditionally relied on deterministic kinetic models, based on ordinary or partial differential equations (ODEs/PDEs), to represent the dynamics of bioprocesses. Classic control approaches for such models, such as feed-forward, feedback, and proportional-integral-derivative control, tend to focus on the short-term impact of actions while ignoring their long-term effects. They also do not distinguish between different sources of uncertainty, and are vulnerable to model misspecification. Furthermore, while these physics-based models have been experimentally shown to be valuable in various types of biomanufacturing processes (Kyriakopoulos et al. 2018, Lu et al. 2015), an appropriate model may simply not be available when dealing with a new biodrug.

These limitations have led to recent interest in models based on Markov decision processes (Peroni et al. 2005, Liu et al. 2013, Martagan et al. 2017, 2019a,b), as well as more sophisticated deep reinforcement learning approaches (Spielberg et al. 2017, Spielberg et al. 2020, Treloar et al. 2020). In a sense, however, these techniques go too far in the opposite direction: they fail to incorporate domain knowledge, and therefore require much greater volumes of training data before they can learn a useful control policy. Additionally, they have limited interpretability, and their ability to handle uncertainty is also limited because they do not consider *model risk*, or error introduced into the policy by misspecification of the stochastic model, resulting from using a small volume of data for calibration. Finally, existing studies of this type are limited to individual unit operations and do not consider the entire process from end to end.

Our proposed framework takes the best from both worlds. Unlike existing papers on RL for process control, which use general-purpose techniques such as neural networks, our approach is model-based: we create a *knowledge graph* that explicitly represents causal interactions within and between different unit operations occurring at different stages of the biomanufacturing process. This is a *hybrid* model, in the sense that it simultaneously incorporates real-world data as well as structural information from existing kinetic models. Using Bayesian statistics, we quantify model uncertainty about the effects of these interactions, where by "uncertainty" we mean not only the stochastic variability inherent in the system, but also the model risk introduced by misspecification. The graph structure of the resulting dynamic Bayesian network (DBN) incorporates information from existing biological, physical, or chemical models: domain experts often know which factors influence each other, and the issue is to precisely measure these effects (the weights of certain edges).

While some of the edge weights in the graph represent the effects of various chemical or physical reactions intrinsic to the biomanufacturing process, others model decisions made in different unit operations (for example, the nutrient feeding strategy for working cells) and thus are directly controllable. The RL problem can then be viewed as a search for a policy that optimally adjusts these weights based on the dynamic state of each unit operation. We find such policies using a novel projected stochastic gradient policy method, which exploits the graph structure to reduce computational cost, by storing and reusing computations made for different paths in the graph. We prove the convergence of this method to a local optimum in the space of policies (the objective is highly non-convex) and give the convergence rate.

We validate our approach in a realistic case study using experimental data for fed-batch fermentation of *Yarrowia lipolytica*, a yeast with numerous biotechnological uses (Bankar et al. 2009). As is typical in the biomanufacturing domain, only eight prior experiments were conducted, and these are the only data available for calibration and optimization of the production process. Nonetheless, by combining these data with information from existing kinetic models, our proposed DBN-RL framework achieves human-level process control with a fairly small number of training iterations, and outperforms the human experts when allowed to run longer. Our approach is far more data-efficient than a state-of-the-art model-free RL method.

In sum, our work makes the following contributions. 1) We propose a Bayesian knowledge graph model which overcomes the limitations of existing process models by explicitly capturing causal interdependencies between factors in the underlying multi-stage stochastic decision process. 2) We develop a model-based reinforcement learning scheme on the Bayesian knowledge graph to find process control policies that are interpretable and robust against model risk, thus mitigating much of the challenge posed by limited process data. 3) We develop an efficient implementation of the RL procedure which exploits the graph structure to allow certain computations to be reused. 4) We demonstrate the efficacy of our approach, against both human experts and a state-of-the-art benchmark, in a case application with real biomanufacturing data. The results show that DBN-RL can be very effective even with a very small amount of process data.

2 Literature Review

Traditionally, biomanufacturing has used physics-based, first-principles models (so-called "kinetic" or "mechanistic" models) of bioprocess dynamics (Mandenius et al. 2013). One example is the work by Liu et al. (2013), which uses an ODE model of cell growth in a fed-batch fermentation process. However, not all bioprocesses are equally well-understood, and first-principles models may simply not be available for certain quality attributes or unit operations

(Rathore et al. 2011). In certain complex operations, such models may oversimplify the process and make a poor fit to real production data (Teixeira et al. 2007).

For this reason, domain experts are increasingly adopting data-driven methods. Such methods are often used when prediction is the main goal: for example, Gunther et al. (2009) aim to predict quality attributes at different stages of a biomanufacturing process using such statistical tools as partial least squares. Statistical techniques can also be used in conjunction with first-principles models, as in Lu et al. (2015), which uses design of experiments to obtain data for a system of ODEs. The main drawback of purely data-driven methods is that they are not easily interpretable, and fail to draw out causal interdependencies between different input factors and process attributes.

Process control, as the name indicates, seeks not only to monitor or predict the process, but to maintain various process parameters within certain acceptable levels in order to guarantee product quality (Jiang and Braatz 2016). Researchers and practitioners have used various standard techniques such as feed-forward, feedback-proportional, and model predictive control (Hong et al. 2018). The work by Lakerveld et al. (2013) shows how such strategies may be deployed at various stages of the production processes and evaluated hierarchically for end-to-end control of an entire pharmaceutical plant. These strategies, however, are usually derived from deterministic first-principles models, and do not consider either stochastic uncertainty or process model risk. Recent work by Martagan et al. (2016, 2017) has sought to address this issue using Markov decision processes, which consider uncertainty to an extent and also allow deeper structural analysis of the optimal control policy. However, these MDP models suffer from the curse of dimensionality (real bioprocesses typically have multidimensional, continuous state and action spaces), and thus focus on a single unit operation under a simplified process model, which exacerbates model risk.

Reinforcement learning (RL) has been shown to attain human-level control in challenging problems in healthcare (Zheng et al. 2021), competitive games (Silver et al. 2016) and other applications. However, these successes were made possible by the availability of large amounts of training data, and the control tasks themselves were well-defined and conducted in a fixed environment. Unfortunately, none of these factors holds in biomanufacturing process control. While there has been some recent work on RL for this domain (Spielberg et al. 2017, Spielberg et al. 2020, Treloar et al. 2020), these methods still require large training datasets; furthermore, the use of general-purpose RL techniques such as deep learning makes it difficult to obtain any interpretable insight from the results. At the same time, such large volumes of data are not only unavailable in biomanufacturing, but strictly speaking should not be *necessary*, because a substantial part of the process dynamics is structured (e.g., according to first-principles models), and one can simply build this structure into the model without having to try to guess it from data.

Our approach captures the strengths of both first-principles and data-driven models through a dynamic Bayesian network hybrid model, which combines both expert knowledge and data (Murphy 2012). The applicability of Bayesian networks to biomanufacturing was first investigated by Xie et al. (2020), which used such a model to capture causal interactions between various unit operations. We build on this work in the present paper, but the model we develop here is much more powerful: first, Xie et al. (2020) infers the interactions from data, whereas we show how to use first-principles models to create a prior; second, Xie et al. (2020) only sought to describe the process, and did not consider the *control* problem to any extent. The core contributions of the present paper are the introduction of a control policy into the Bayesian network, and the optimization of this policy using gradient ascent. These developments require considerable new developments in modeling, computation, and theory, all of which are new to this paper.

3 Modeling Biomanufacturing Processes Using Dynamic Bayesian Networks

In this section, we show how one can create a dynamic Bayesian network (DBN) model for biomanufacturing decision processes. Section 3.1 shows how the network structure and dynamics can be extracted from domain-specific kinetic models (when such models are available). Section 3.2 shows how the information obtained from kinetic models can be represented using linear Gaussian dynamics, and further improved using Bayesian updating with experimental data. Section 3.3 augments this model with a linear control policy which is evaluated using a linear reward function. Finally, Section 3.4 presents structural results that will be used in later algorithmic developments.

3.1 Extracting Graph Structure From Kinetic Models

A typical biomanufacturing process consists of distinct unit operations (cell culture, production, purification etc.) which take place sequentially. The state of the process includes "critical quality attributes" (CQAs) such as biomass and impurity concentrations. The actions include "critical process parameters" (CPPs) such as temperature and feed rate. The decision-maker monitors the CQAs using sensors or lab tests, and adjusts the CPPs as necessary. Both CPPs and CQAs exert causal effects on future CQAs. Furthermore, CQAs at any stage may also be influenced by uncontrolled

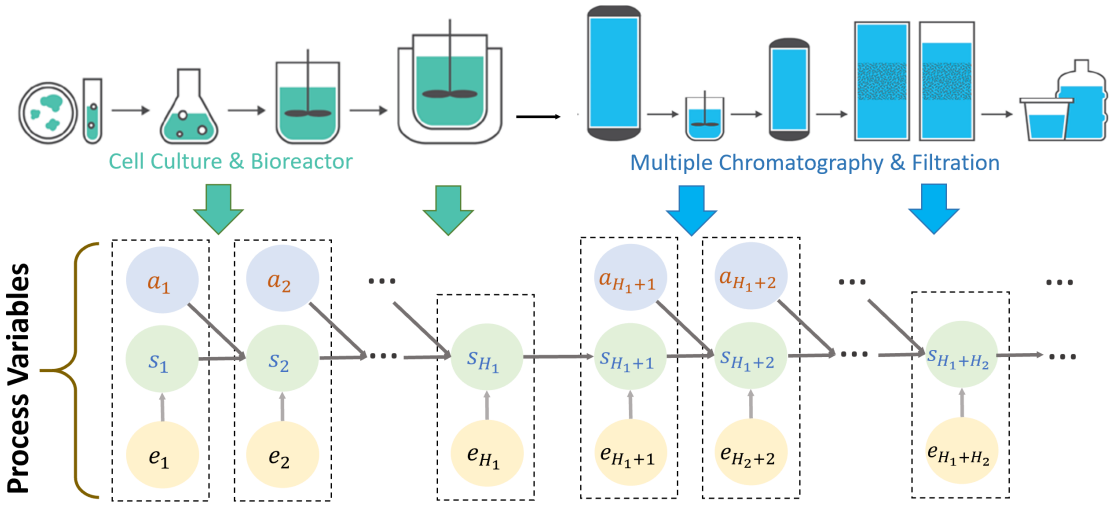


Figure 1: Illustrative example of network model for an integrated biopharmaceutical production process. We use $(\boldsymbol{s}_t, \boldsymbol{a}_t)$ to represent CQAs/CPPs at time period t. Directed edges indicate causal dependencies.

(exogenous) factors such as contamination. All of these effects are represented as edges in an interpretable relational graph, with CPPs and CQAs being nodes; Figure 1 gives an illustrative example.

The quantity s_t^k (respectively, a_t^k) denotes the value of the kth CQA node (kth CPP node) at time t. In the language of Markov decision processes, s_t^k is the kth dimension of the state variable, while a_t^k is the kth dimension of the action or decision variable. We suppose that there are n CQAs and m CPPs in each time period (these quantities could be time-dependent, but for notational simplicity we keep them constant in our presentation), and that there are H time periods in all. We use the notation $\boldsymbol{s}_t = \left\{s_t^1, ..., s_t^n\right\}$ and $\boldsymbol{a}_t = \left\{a_t^1, ..., a_t^m\right\}$ to denote the state and action variables at time period t.

The biomanufacturing literature generally uses kinetic models based on PDEs or ODEs to model the dynamics of these variables. Suppose that $\boldsymbol{s}_t$ evolves according to the ordinary differential equation

$$\frac{d\boldsymbol{s}}{dt} = \boldsymbol{f}(\boldsymbol{s}, \boldsymbol{a}) \tag{1}$$

where $\boldsymbol{f}(\cdot) = (f_1, f_2, \ldots, f_n)$ encodes the causal interdependencies between various CPPs and CQAs. One typically assumes that the functional form of $\boldsymbol{f}$ is known, though it may also depend on additional parameters that are calibrated from data. Supposing that the bioprocess is monitored on a small time scale using sensors, let us replace (1) by the first-order Taylor approximation

$$\frac{\Delta \boldsymbol{s}_{t+1}}{\Delta t} = \boldsymbol{f}(\boldsymbol{\mu}_t^s, \boldsymbol{\mu}_t^a) + J_f^s(\boldsymbol{\mu}_t^s)(\boldsymbol{s}_t - \boldsymbol{\mu}_t^s) + J_f^a(\boldsymbol{\mu}_t^a)(\boldsymbol{a}_t - \boldsymbol{\mu}_t^a), \tag{2}$$

where $\Delta \boldsymbol{s}_{t+1} = \boldsymbol{s}_{t+1} - \boldsymbol{s}_t$, and J_f^s, J_f^a denote the Jacobian matrices of $\boldsymbol{f}$ with respect to $\boldsymbol{s}_t$ and $\boldsymbol{a}_t$, respectively. The interval Δt can change with time. For notation simplification, we keep it constant. The approximation is taken at a point $(\boldsymbol{\mu}_t^s, \boldsymbol{\mu}_t^a)$ to be elucidated later. We can then rewrite (2) as

$$\boldsymbol{s}_{t+1} = \boldsymbol{\mu}_t^s + \Delta t \cdot \boldsymbol{f}(\boldsymbol{\mu}_t^s, \boldsymbol{\mu}_t^a) + (\Delta t \cdot J_f(\boldsymbol{\mu}_t^s) + 1)(\boldsymbol{s}_t - \boldsymbol{\mu}_t^s) + \Delta t \cdot J_f(\boldsymbol{\mu}_t^a)(\boldsymbol{a}_t - \boldsymbol{\mu}_t^a) + R_{t+1}, \tag{3}$$

where R_{t+1} is a remainder term modeling the effect from other uncontrolled factors. In this way, the original process dynamics have been linearized, with R_{t+1} serving as a residual. One can easily represent (3) using a network model. An edge exists from s_t^k (respectively, a_t^k) to s_{t+1}^l if the (k, l)th entry of J_f^s (respectively, J_f^a) is not identically zero. As will be seen shortly, the linearized dynamics can provide prior knowledge to the state transition model of the dynamic Bayesian network.

We should note that the Bayesian network approach does not *require* a preexisting kinetic model. In the early stages of process development, the practitioner may not have sufficient knowledge of the bioprocess to design a kinetic model. In that case, we can still estimate a linear statistical model from $\{\boldsymbol{s}_t, \boldsymbol{a}_t\}$ to $\boldsymbol{s}_{t+1}$ to quantify the main effects. However, if a kinetic model is available, our framework can leverage it to extract problem structure, thus helping to mitigate the challenge of small sample sizes.

3.2 Bayesian Networks With Linear Gaussian Dynamics

We now consider a given network structure and describe a model where each CQA variable s_t^k evolves according to linear Gaussian dynamics. Bayesian updating will then be used to learn the model parameters from data. We first model $a_t^k \sim \mathcal{N}(\lambda_t^k, (\sigma_t^k)^2)$ for each CPP a_t^k. Note that, for the moment, we are not explicitly modeling $\boldsymbol{a}_t$ as a decision variable, even though in practice it is controlled by the decision-maker. Instead, we treat $\boldsymbol{a}_t$ as a random variable, for two main reasons. First, the kinetic models used in biomanufacturing focus on modeling the bioprocessing dynamics, rather than the control. Second, in practice, the specifications of the production process (which ensure that the final product meets quality requirements) treat CPPs as ranges of values, within which variation is possible. The problem of actually choosing a control policy will be revisited in Section 3.3.

Similarly, we suppose that $s_1^k \sim \mathcal{N}(\mu_1^k, (v_1^k)^2)$, modeling variation in the initial state because CQAs (e.g., of seeding cells and serum-based media) are not completely controllable. The values of CQAs in subsequent time periods are determined by

$$s_{t+1}^k = \mu_{t+1}^k + \sum_{X_t^j \in Pa(s_{t+1}^k)} \beta_t^{jk}(X_t^j - \mu_t^j) + e_{t+1}^k, \qquad t > 1, k = 1, ..., n, \tag{4}$$

where $Pa(s_t^k)$ denotes the set of parent nodes of s_t^k in the network, and $e_{t+1}^k \sim \mathcal{N}\left(0, \left(v_{t+1}^k\right)^2\right)$ is an independent residual noise term.

Then, the distribution of the entire trajectory $\boldsymbol{\tau} = (\boldsymbol{s}_1, \boldsymbol{a}_1, \boldsymbol{s}_2, \boldsymbol{a}_2, \ldots, \boldsymbol{s}_T)$ of the stochastic process can be represented by a product

$$p(\boldsymbol{\tau}) \;=\; \prod_{t=1}^{H}\left[\prod_{k=1}^{m}\mathcal{N}(\lambda_t^k, (\sigma_t^k)^2)\prod_{k=1}^{n}\mathcal{N}\left(\mu_{t+1}^k + \sum_{X_t^j \in Pa(s_{t+1}^k)} \beta_t^{jk}(X_t^j - \mu_t^j), (v_{t+1}^k)^2\right)\right]$$

of conditional distributions.

In the following, we assume that $Pa(s_{t+1}^k) = \{\boldsymbol{s}_t, \boldsymbol{a}_t\}$ to avoid complicating the notation, but there is no difficulty in considering a smaller set of parent nodes. Let $\boldsymbol{\beta}_t^s$ be the $n \times n$ matrix whose (j,k)th element is the linear coefficient β_t^{jk} in (4) corresponding to the edge from state $\boldsymbol{s}_t$ to the next state $\boldsymbol{s}_{t+1}$. Similarly, let $\boldsymbol{\beta}_t^a$ be the $m \times n$ matrix of analogous coefficients corresponding to the edges from action $\boldsymbol{a}_t$ to the next state $\boldsymbol{s}_{t+1}$. Then, the dynamics (4) can be rewritten in matrix form,

$$\boldsymbol{s}_{t+1} = \boldsymbol{\mu}_{t+1}^s + (\boldsymbol{\beta}_t^s)^\top (\boldsymbol{s}_t - \boldsymbol{\mu}_t^s) + (\boldsymbol{\beta}_t^a)^\top (\boldsymbol{a}_t - \boldsymbol{\mu}_t^a) + V_{t+1}\boldsymbol{z}_{t+1}, \tag{5}$$

where $\boldsymbol{z}_{t+1}$ is a n-dimensional standard normal random vector, $V_{t+1} \triangleq \text{diag}(v_{t+1}^k)$ is a diagonal matrix of residual variances, and $\boldsymbol{\mu}_t^s = (\mu_t^1, \ldots, \mu_t^n)$, $\boldsymbol{\mu}_t^a = (\lambda_t^1, \ldots, \lambda_t^m)$. Letting $\boldsymbol{\sigma}_t = (\sigma_t^1, \ldots, \sigma_t^m)$ and $\boldsymbol{v}_t = (v_t^1, \ldots, v_t^n)$, the list of parameters for the entire model can be denoted by $\boldsymbol{w} = (\boldsymbol{\mu}^s, \boldsymbol{\mu}^a, \boldsymbol{\beta}, \boldsymbol{\sigma}, \boldsymbol{v}) = \{(\boldsymbol{\mu}_t^s, \boldsymbol{\mu}_t^a, \boldsymbol{\beta}_t^s, \boldsymbol{\beta}_t^a, \boldsymbol{\sigma}_t, \boldsymbol{v}_t) | 0 \le t \le H\}$ where $\boldsymbol{\beta} = (\boldsymbol{\beta}^a, \boldsymbol{\beta}^s)$.

It is easy to see that (5) has the same form as (3) if we let

$$\begin{aligned} \boldsymbol{\mu}_{t+1}^s &= \boldsymbol{\mu}_t^s + \Delta t \cdot \boldsymbol{f}(\boldsymbol{\mu}_t^s, \boldsymbol{\mu}_t^a), \\ \boldsymbol{\beta}_t^s &= \Delta t \cdot J_f(\boldsymbol{\mu}_t^s) + 1, \\ \boldsymbol{\beta}_t^a &= \Delta t \cdot J_f(\boldsymbol{\mu}_t^a), \end{aligned}$$

and treat R_{t+1} as the residual. We then obtain data $\mathcal{D} = \left\{\boldsymbol{\tau}^{(n)}\right\}_{n=1}^R$ and quantify the estimation uncertainty for the model parameters using the posterior distribution $p(\boldsymbol{w}|\mathcal{D})$. We can generate the posterior samples from it using the method of Gibbs sampling (Gelfand 2000). Our implementation of this technique closely follows Xie et al. (2020), so we omit the details to avoid distracting the reader; a brief description is given in Appendix 8.

It is worth noting that the data $\mathcal{D} = \left\{\boldsymbol{\tau}^{(n)}\right\}_{n=1}^R$ may be obtained in different ways. If a reliable kinetic model of the underlying bioprocess is available, we can first calibrate this model using real experimental data, and then simulate many trajectories $\boldsymbol{\tau}^{(n)}$ from it using different initializations and controls. We can use these simulated data to get a better fit of the Bayesian network to the kinetic model, thus incorporating more of the rich structural information that the latter provides, and helping to mitigate the problem of small sample sizes. However, if such a model is not available, $\mathcal{D}$ may be limited only to the existing experimental data. Either way, we perform the same statistical inference procedure on $\mathcal{D}$.

3.3 Linear Rewards and Policies

The process trajectory $\boldsymbol{\tau}$ is evaluated in terms of revenue (based on how much bioproduct was harvested at the end of the process) as well as cost. The latter includes the fixed cost of operating the bioreactor, running the sensors, and maintaining the facilities, as well as variable manufacturing costs related to raw materials, labor, quality control, and purification. Variable costs depend on the values of the CPPs and CQAs across the entire production process: for example, purification costs depend on the quantity of unwanted metabolic wastes, which can be one of the CQAs evolving over time, while manufacturing costs depend on CPPs such as the amount of raw materials used.

The biomanufacturing industry often uses a linear reward structure, e.g., $r_t\left(\boldsymbol{s}_t, \boldsymbol{a}_t\right) = m_t + \boldsymbol{b}_t^\top \boldsymbol{a}_t + \boldsymbol{c}_t^\top \boldsymbol{s}_t$, where m_t is the fixed cost at time t, $\boldsymbol{b}_t$ is the variable manufacturing cost, and $\boldsymbol{c}_t$ combines both purification cost and product revenue. The trajectory $\boldsymbol{\tau}$ is influenced, not only by the actions of the controller, but also by the underlying uncertain model parameters $\boldsymbol{w}$, which represent the intrinsic attributes of the bioprocess. Our eventual goal is to use the reward function to guide control policies; however, we are only interested in control of models whose parameters $\boldsymbol{w}$ fall into some realistic range. Bioprocesses are subject to certain physical and biochemical laws: for example, in fermentation, cell growth rate and oxygen/substrate uptake rate generally do not fall outside a certain range. We let $\mathcal{W}$ be the set of all model parameters $\boldsymbol{w}$ that satisfy these fundamental laws, and modify the reward structure to be

$$r_t(\boldsymbol{s}_t, \boldsymbol{a}_t; \boldsymbol{w}) = \begin{cases} m_t + \boldsymbol{b}_t^\top \boldsymbol{a}_t + \boldsymbol{c}_t^\top \boldsymbol{s}_t & \boldsymbol{w} \in \mathcal{W}, \\ m_c, & \boldsymbol{w} \notin \mathcal{W}, \end{cases} \tag{6}$$

where m_c is a negative constant. Essentially this means that, if the underlying model is outside the range of interest, then it does not matter how we set CPPs. This will prevent us from training control policies on unrealistic dynamics. To streamline the notation, we often omit the explicit dependence of r_t on $\boldsymbol{w}$ in the following, except when necessary.

We can now formally define control policies and the optimization problem solved in this work. At each time t, the decisions are set according to $\boldsymbol{a}_t = \pi_t\left(\boldsymbol{s}_t\right)$, where π_t maps the state vector $\boldsymbol{s}_t$ into the space of all possible action values. The policy $\pi = \{\pi_t\}_{t=1}^H$ is the collection of these mappings over the entire planning period. Let

$$J(\pi; \boldsymbol{w}) = \mathbb{E}_{\boldsymbol{\tau}}\left[\sum_{t=1}^{H} r_t(\boldsymbol{s}_t, \boldsymbol{a}_t) \middle| \boldsymbol{\pi}, \boldsymbol{s}_1, \boldsymbol{w} \right] \tag{7}$$

be the expected cumulative reward earned by policy π during the planning period. The expected value is taken over the *conditional* distribution of the process trajectory $\boldsymbol{\tau}$ given the model parameters $\boldsymbol{w}$. Thus, the same policy will perform differently under different models. Ideally, given a set $\mathcal{P}$ of candidate policies, we would like to find $\arg\max_{\pi \in \mathcal{P}} J\left(\pi; \boldsymbol{w}^{\text{true}}\right)$, where $\boldsymbol{w}^{\text{true}}$ contains the true parameters describing the underlying bioprocess. However, the true model is unknown, and optimizing with respect to any fixed $\boldsymbol{w}$ runs the risk of poor performance due to model misspecification. In other words, the optimal policy with respect to $\boldsymbol{w}$ may be very suboptimal with respect to $\boldsymbol{w}^{\text{true}}$, especially under the situations with very limited real-world experimental data. To mitigate this *model risk*, we instead search for a policy that performs well, on average, across many different posterior samples of process model with $\boldsymbol{w} \sim p\left(\boldsymbol{w}|\mathcal{D}\right)$. Formally, we wish to solve

$$\pi^* = \arg\max_{\pi \in \mathcal{P}} \mathcal{J}\left(\pi_{\boldsymbol{\theta}}\right) \tag{8}$$

where

$$\mathcal{J}\left(\pi_{\boldsymbol{\theta}}\right) = \mathbb{E}_{\boldsymbol{w} \sim p(\boldsymbol{w}|\mathcal{D})}\left[J\left(\pi_{\boldsymbol{\theta}}; \boldsymbol{w}\right)\right], \tag{9}$$

with the expectation taken over the posterior distribution of $\boldsymbol{w}$ given the real experimental data $\mathcal{D}$. Recall from (7) that $J\left(\pi; \boldsymbol{w}\right)$ is an expected value over the stochastic uncertainty in the bioprocess, i.e., the uncertainty inherent in the trajectory $\boldsymbol{\tau}$. Equation (9) takes an additional expectation to account for model risk, thus hedging against this additional source of uncertainty.

Although the expectation in (9) cannot be computed in closed form, even for a fixed policy π, one could potentially estimate it empirically by averaging over multiple samples from the posterior $p\left(\boldsymbol{w}|\mathcal{D}\right)$. As was discussed in Section 3.2, Gibbs sampling can be used to generate such samples. The complexity of this computation, however, makes it difficult to optimize over all possible policies, especially since each π_t is defined on the set of all possible (continuous) CQA values. The intractability of this problem is well-known as the "curse of dimensionality" (Powell 2011). To make the problem more tractable, we impose the parametric linear structure

$$\pi_{\boldsymbol{\vartheta}_t}(\boldsymbol{s}_t) = \boldsymbol{\mu}_t^a + \boldsymbol{\vartheta}_t^\top (\boldsymbol{s}_t - \boldsymbol{\mu}_t^s), \tag{10}$$

where $\boldsymbol{\mu}_t^a$ is the mean action value and $\boldsymbol{\vartheta}_t$ is an $n \times m$ matrix of coefficients. To be consistent with the model of Section 3.2, we use $\boldsymbol{\mu}_t^a = \left(\lambda_t^1, ..., \lambda_t^m\right)$. The linear policy $\pi_{\boldsymbol{\vartheta}}$ is thus characterized by $\boldsymbol{\theta} = \{\boldsymbol{\vartheta}_t\}_{t=1}^{H-1}$, and the set $\mathcal{P}$ of possible

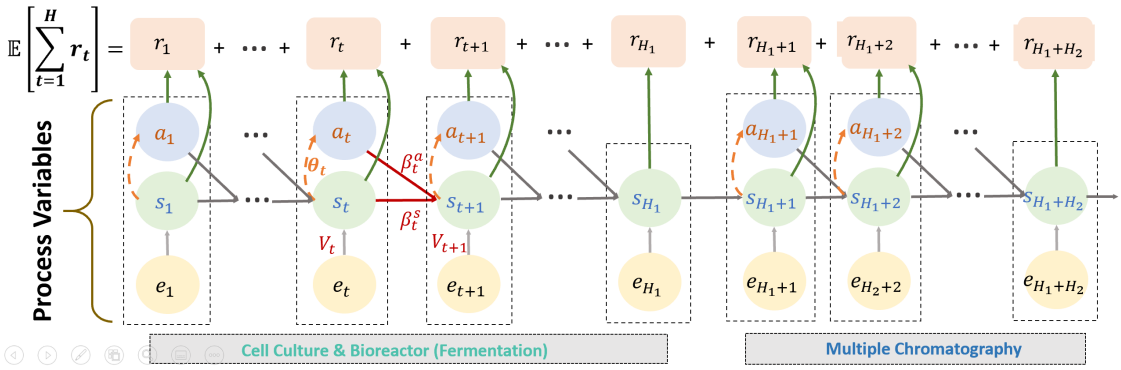


Figure 2: Illustrative example of policy augmented Bayesian network (PABN).

policies can be replaced by a suitable closed convex set $\mathbb{C}$ of candidate values for the parameters $\boldsymbol{\vartheta}$. The remainder of this paper is devoted to the problem of optimizing these policy parameters.

In practice, the assumption of a linear policy is reasonable if the process is monitored and controlled on a sufficiently small time scale. This is because, on such a time scale, the effects of CQAs on CPPs are monotonic and approximately linear. For example, higher cell growth rate is typically linearly associated with higher feeding rate in the cell culture growth phase.

With the introduction of rewards and policies, the network structure is modified from its original form as laid out in Sections 3.1-3.2. An illustrative example is given in Figure 2. As in Figure 1, the integrated biomanufacturing process consists of fermentation for protein production followed by chromatograph purification. As before, the network models the causal effects of $\{\boldsymbol{s}_t, \boldsymbol{a}_t\}$ on future states $\boldsymbol{s}_{t'}$ in later stages $t' > t$, but now includes additional edges connecting CQAs (states) to CPPs (actions) at the same time stage, representing the causal effect of the policy. We use the term *policy augmented Bayesian network (PABN)* to refer to this structure.

To our knowledge, PABNs are a new modeling concept. Structurally, they have certain similarities to neural networks, in that we are choosing parameters to optimize outputs of the network, but a major difference is that each time period produces its own output, and the goal is to maximize cumulative reward. Furthermore, the output is an economic quantity, not a prediction, and the dynamics connecting nodes at different time periods are an interpretable model of an actual physical process, not simply a black-box transformation of data.

3.4 Structural Properties

We close this section with two important computational results which will be used later to facilitate policy optimization. The first result relates any CQA state $\boldsymbol{s}_{t+1}$ at any time $t+1$ to an earlier value $\boldsymbol{s}_h$ with $h \leq t$. This is useful because, as shown in Figure 2, there may be multiple paths connecting a given pair of nodes. For example, $\boldsymbol{s}_t$ is connected to $\boldsymbol{s}_{t+1}$ directly through the coefficient $\boldsymbol{\beta}_t^s$, but also through the CPP action $\boldsymbol{a}_t$ via the coefficients $\boldsymbol{\beta}_t^a \boldsymbol{\vartheta}_t$. Thus, $\boldsymbol{s}_{t+1}$ depends on $\boldsymbol{s}_h$ through all possible pathways connecting the two. The nature of these dependence is characterized in the following proposition, whose detailed derivation is given in Appendix 10.1.

Proposition 1 *For any given decision epoch t and starting state $\boldsymbol{s}_h$ with $h \in \{1, 2, \ldots, t\}$, we have*

$$\boldsymbol{s}_{t+1} = \boldsymbol{\mu}_{t+1}^s + \mathbf{R}_{h,t}(\boldsymbol{s}_h - \boldsymbol{\mu}_h^s) + \left[\sum_{i=h}^{t} \mathbf{R}_{i,t} V_i \boldsymbol{z}_i + V_{t+1}\boldsymbol{z}_{t+1}\right],$$

where V_i and $\boldsymbol{z}_i$ are as in (5), and $\mathbf{R}_{i,t} = \prod_{j=i}^{t}\left[\left(\boldsymbol{\beta}_j^s\right)^\top + \left(\boldsymbol{\beta}_j^a\right)^\top \boldsymbol{\vartheta}_j^\top\right]$ represents the product of pathway coefficients from time step i to t.

Using Proposition 1, we can derive an explicit expression for the objective value $J(\pi_{\boldsymbol{\theta}}; \boldsymbol{w})$ of the linear policy $\pi_{\boldsymbol{\theta}}$, as well as the mean and variance of the state vector $\boldsymbol{s}_{t+1}$ under such a policy. Again, the detailed derivation can be found in Appendix 10.2. From this result, we can see that the objective function is highly non-convex in the policy parameters $\boldsymbol{\theta}$.

Proposition 2 *Fix a model $\boldsymbol{w} = (\boldsymbol{\mu}^s, \boldsymbol{\mu}^a, \boldsymbol{\beta}, \boldsymbol{\sigma}, \boldsymbol{v})$, a parametric linear policy $\pi_{\boldsymbol{\theta}}$, and an initial state vector $\boldsymbol{s}_1$. Then:*

1. *The mean and variance of $\boldsymbol{s}_{t+1}$ are given by*

$$\begin{aligned}\mathbb{E}[\boldsymbol{s}_{t+1}] &= \boldsymbol{\mu}^s_{t+1} + \mathbf{R}_{h,t}(\mathbb{E}[\boldsymbol{s}_h] - \boldsymbol{\mu}^s_h) = \boldsymbol{\mu}^s_{t+1} + \mathbf{R}_{1,t}(\boldsymbol{s}_1 - \boldsymbol{\mu}^s_1) \quad \textit{for } h \in \{1, 2, \ldots, t\} \\ \textit{Var}[\boldsymbol{s}_{t+1}] &= \sum_{h=1}^{t} \mathbf{R}_{h,t} V_h \mathbf{R}_{h,t}^\top + V_{t+1}.\end{aligned}$$

2. *Under the linear reward structure of (6),*

$$J\left(\pi_{\boldsymbol{\theta}}; \boldsymbol{w}\right) = \begin{cases} \sum_{t=1}^{H} \left\{ m_t + \boldsymbol{b}_t^\top \boldsymbol{\mu}_t^a + \boldsymbol{c}_t^\top \boldsymbol{\mu}_t^s + (\boldsymbol{b}_t^\top \boldsymbol{\vartheta}_t^\top + \boldsymbol{c}_t^\top) \mathbf{R}_{1,t}(\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s) \right\} & \boldsymbol{w} \in \mathcal{W}, \\ H m_c & \boldsymbol{w} \notin \mathcal{W}. \end{cases} \tag{11}$$

4 Policy Optimization With Projected Stochastic Gradients

In this section, we propose a policy optimization method to solve (8) with $\mathcal{P}$ restricted to the class of parametric linear policies introduced in (10). Essentially, we use gradient ascent to optimize $\mathcal{J}$ over the space of parametric linear policies, but there are some nuances in the analysis due to the fact that the space of policies is constrained to the set $\mathbb{C}$. Section 4.1 discusses the computation of the policy gradient, while Section 4.2 states the overall optimization framework in which the gradient computation is used.

4.1 Gradient Estimation and Computation

For notational convenience, we represent the parametric linear policy $\pi_{\boldsymbol{\theta}}$ by the parameter vector $\boldsymbol{\theta}$, and write $J, \mathcal{J}$ in (7) and (8) as functions of $\boldsymbol{\theta}$. We first establish the differentiability of the objective function $\mathcal{J}$, which will allow us to use gradient ascent to optimize the parameters. In the following, we understand $\boldsymbol{\theta}$ as a vector, i.e., $\boldsymbol{\theta} = \left(\text{vec}\left(\boldsymbol{\vartheta}_1\right), ..., \text{vec}\left(\boldsymbol{\vartheta}_{H-1}\right)\right)^\top$, where $\text{vec}\left(\cdot\right)$ denotes a linear transformation converting an $n \times m$ matrix into a column vector. For notation simplification, we write the policy gradient $\nabla_{\boldsymbol{\theta}} \mathcal{J}\left(\boldsymbol{\theta}\right)$ as $\nabla \mathcal{J}\left(\boldsymbol{\theta}\right)$ in the following presentation.

Lemma 1 *The objective function $\mathcal{J}$ is differentiable and its gradient satisfies*

$$\nabla \mathcal{J}\left(\boldsymbol{\theta}\right) = \mathbb{E}_{\boldsymbol{w} \sim p(\boldsymbol{w}|\mathcal{D})}\left[\nabla_{\boldsymbol{\theta}} J\left(\boldsymbol{\theta}; \boldsymbol{w}\right)\right]. \tag{12}$$

Furthermore, $\nabla \mathcal{J}$ is L-smooth over the closed convex set $\mathbb{C}$, that is,

$$\|\nabla \mathcal{J}(\boldsymbol{x}) - \nabla \mathcal{J}(\boldsymbol{y})\| \leq L\|\boldsymbol{x} - \boldsymbol{y}\|, \qquad \boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}.$$

Equation (12) justifies the interchange of derivative and expectation, allowing us to focus on computing the gradient $\nabla_{\boldsymbol{\theta}} J\left(\boldsymbol{\theta}; \boldsymbol{w}\right)$. The outer expectation over the posterior distribution of $\boldsymbol{w}$ can be estimated using sample average approximation (SAA) method: we use Gibbs sampling, as discussed in Section 3.2, to generate posterior samples $\left\{\boldsymbol{w}^{(b)}\right\}_{b=1}^{B}$ from the distribution $p\left(\boldsymbol{w}|\mathcal{D}\right)$, and calculate

$$\nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}) = \frac{1}{B} \sum_{b=1}^{B} \nabla_{\boldsymbol{\theta}} J\left(\boldsymbol{\theta}; \boldsymbol{w}^{(b)}\right). \tag{13}$$

In the following discussion, we consider a fixed $h \in \{1, ..., H-1\}$ as well as a fixed model $\boldsymbol{w} \in \mathcal{W}$. Recalling (7), we write

$$\nabla_{\boldsymbol{\vartheta}_h} J(\boldsymbol{\theta}; \boldsymbol{w}) = \sum_{t=1}^{H} \nabla_{\boldsymbol{\vartheta}_h} \mathbb{E}_{\boldsymbol{\tau}}\left[r_t(\boldsymbol{s}_t, \boldsymbol{a}_t) \middle| \boldsymbol{\pi}_\theta, \boldsymbol{s}_1, \boldsymbol{w}\right].$$

By plugging in the reward and policy functions in (6) and (10), the expected reward becomes,

$$\mathbb{E}_{\boldsymbol{\tau}}\left[\sum_{t=1}^{H} r_t(\boldsymbol{s}_t, \boldsymbol{a}_t) \middle| \boldsymbol{\pi}_\theta, \boldsymbol{s}_1, \boldsymbol{w}\right] = m_t + \boldsymbol{c}_t^\top \mathbb{E}[\boldsymbol{s}_t | \boldsymbol{\pi}_{\boldsymbol{\theta}}, \boldsymbol{s}_1, \boldsymbol{w}] + \boldsymbol{b}_t^\top \left(\boldsymbol{\mu}_t^a + \boldsymbol{\vartheta}_t^\top (\mathbb{E}[\boldsymbol{s}_t | \boldsymbol{\pi}_{\boldsymbol{\theta}}, \boldsymbol{s}_1, \boldsymbol{w}] - \boldsymbol{\mu}_t^s)\right).$$

Similarly, for $\boldsymbol{w} \notin \mathcal{W}$, we have $\mathbb{E}_{\boldsymbol{\tau}}\left[\sum_{t=1}^{H} r_t(\boldsymbol{s}_t, \boldsymbol{a}_t) \middle| \boldsymbol{\pi}_\theta, \boldsymbol{s}_1, \boldsymbol{w}\right] = H m_c$.

Let $\bar{\boldsymbol{s}}_t = \mathbb{E}[\boldsymbol{s}_t | \boldsymbol{\pi}_{\boldsymbol{\theta}}, \boldsymbol{s}_1, \boldsymbol{w}]$. Using Propositions 1-2 and the functional forms of the reward and policy functions, we obtain an expression

$$\bar{r}_t(\boldsymbol{\theta}; \boldsymbol{w}) \equiv \mathbb{E}_{\boldsymbol{\tau}}\left[r_t(\boldsymbol{s}_t, \boldsymbol{a}_t) | \boldsymbol{\pi}_{\boldsymbol{\theta}}, \boldsymbol{s}_1, \boldsymbol{w}\right] = m_t + \boldsymbol{c}_t^\top \bar{\boldsymbol{s}}_t + \boldsymbol{b}_t^\top (\boldsymbol{\mu}_t^a + \boldsymbol{\vartheta}_t^\top (\bar{\boldsymbol{s}}_t - \boldsymbol{\mu}_t^s)) \tag{14}$$

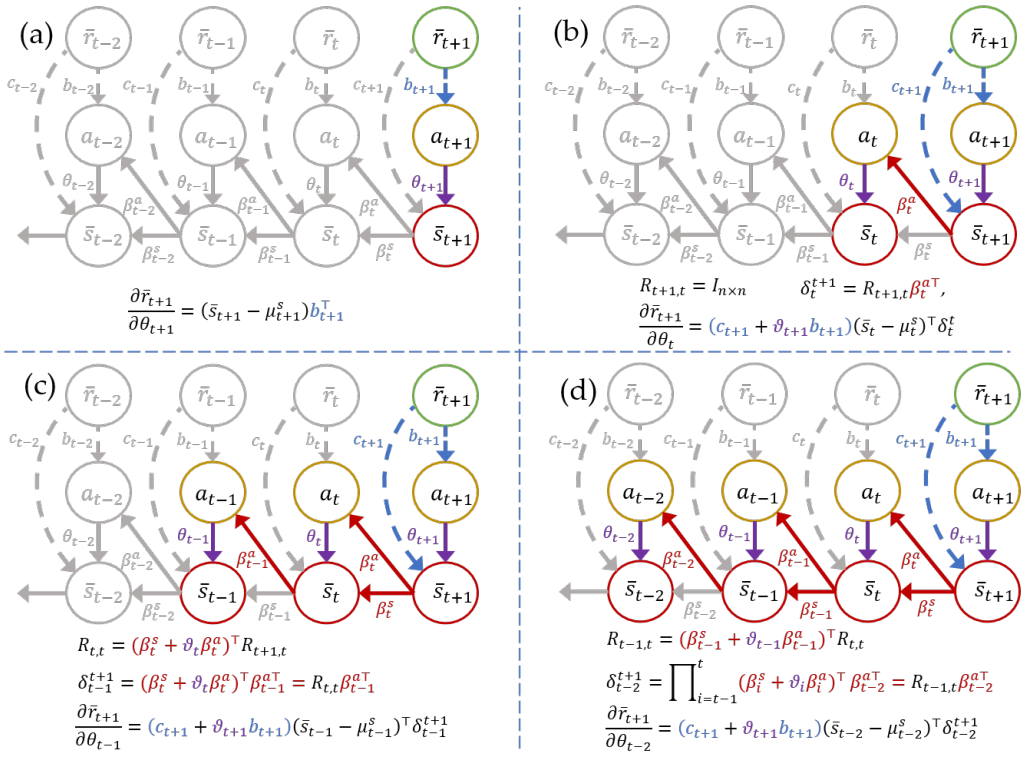


Figure 3: Illustration showing nested backpropagation computations, highlighting the pathways involved in each policy gradient calculation.

for the expected reward at time t. This expression is equivalent to the one obtained in Proposition 2, but functionally it serves a different purpose: instead of expanding out all the pathways leading from time 1 to time t, we instead represent them by $\bar{\boldsymbol{s}}_t$. We can also write a partial expansion of the pathways starting at some earlier time $h \leq t$. Such representations are crucial in policy optimization, where we need to evaluate partial derivatives of $\bar{r}_t$ with respect to $\boldsymbol{\vartheta}_h$ for just such an earlier time h. Thus, (14) leads directly into the following computational result; the proof is given in Appendix 10.3.

Theorem 1 (Nested Backpropagation) *Let $\mathbf{R}_{h,t}$ be as in Proposition 1, with $\mathbf{R}_{t,t-1} = \mathbb{I}_{n\times n}$ by convention. Fix a model $\boldsymbol{w}$ and policy parameter vector $\boldsymbol{\theta}$. For any $h \leq t$, we have*

$$\frac{\partial \bar{r}_t(\boldsymbol{\theta}; \boldsymbol{w})}{\partial \boldsymbol{\vartheta}_h} = \begin{cases} (\boldsymbol{c}_t + \boldsymbol{\vartheta}_t \boldsymbol{b}_t)(\bar{\boldsymbol{s}}_h - \boldsymbol{\mu}^s_h)^\top \boldsymbol{\delta}^t_h & \text{if } h < t \\ (\bar{\boldsymbol{s}}_h - \boldsymbol{\mu}^s_h)\boldsymbol{b}^\top_t, & \text{if } h = t \end{cases} \tag{15}$$

where $\boldsymbol{\delta}^t_h = \mathbf{R}_{h+1,t-1}\,(\boldsymbol{\beta}^a_h)^\top$.

Theorem 1 gives us a computationally efficient way to update the gradient of $J(\boldsymbol{\theta}; \boldsymbol{w})$ which we call "*nested backpropagation*" to highlight both similarities and differences with the classic backpropagation algorithm used in the calibration of neural networks (Goh 1995). Each reward can be backpropagated to any policy parameter (say, $\boldsymbol{\vartheta}_h$) through the network or knowledge graph as shown in Figure 3. The term "backpropagation" refers to the fact that the calculation of gradients proceeds backward through the network, from time t to time h. This structure is shared by neural networks. The key distinction, however, is that in our problem we must compute the gradient of $\bar{r}_t$ for *every* time period t, whereas in classic neural networks there is a single output function calculated at the final (terminal) node. Thus, there may be overlapping pathways between two such gradients, creating opportunities to save and reuse gradient computations. Specifically, the computation of $\frac{\partial \bar{r}_t}{\partial \boldsymbol{\vartheta}_h}$ can reuse the propagation pathways $\mathbf{R}_{h+1,t-1}$ through applying the equation

$$\mathbf{R}_{h,t-1} = (\boldsymbol{\beta}^s_h + \boldsymbol{\vartheta}_h \boldsymbol{\beta}^a_h)^\top \mathbf{R}_{h+1,t-1}.$$

Figure 3 illustrates these reused computations through the examples (a) $\frac{\partial \bar{r}_{t+1}}{\partial \boldsymbol{\vartheta}_{t+1}}$, (b) $\frac{\partial \bar{r}_{t+1}}{\partial \boldsymbol{\vartheta}_t}$, (c) $\frac{\partial \bar{r}_{t+1}}{\partial \boldsymbol{\vartheta}_{t-1}}$ and (d) $\frac{\partial \bar{r}_{t+1}}{\partial \boldsymbol{\vartheta}_{t-2}}$. In Figure 3(a), the gradient $\frac{\partial \bar{r}_{t+1}}{\partial \boldsymbol{\vartheta}_{t+1}}$ is computed using only the nodes r_{t+1}, $\boldsymbol{a}_{t+1}$ and $\boldsymbol{s}_{t+1}$. In Figure 3(b), the gradient

Algorithm 1: Nested backpropagation procedure for policy gradient computation.

Input: DBN coefficients $\boldsymbol{w}$, policy parameters $\boldsymbol{\vartheta}_t$, reward function $r_t(\bar{\boldsymbol{s}}_t, \boldsymbol{a}_t) = m_t + \boldsymbol{c}_t\bar{\boldsymbol{s}}_t + \boldsymbol{b}_t\boldsymbol{a}_t$ for all t;
1. Compute $\mathbf{R}_{h,t}$ and $\boldsymbol{\delta}_h^t$ with $t = 1, 2, \ldots, H$ and $h = 1, 2, \ldots, t$ as follows:
for $t = 1, 2, \ldots, H$ **do**
  Set $\mathbf{R}_{t+1,t} = I_{n\times n}$;
  for $h = t, t-1, \ldots, 1$ **do**
    (a) $\boldsymbol{\delta}_h^t = \mathbf{R}_{h+1,t}\boldsymbol{\beta}_h^{a\top}$;
    (b) $\mathbf{R}_{h,t} = (\boldsymbol{\beta}_h^s + \boldsymbol{\vartheta}_h\boldsymbol{\beta}_h^a)^\top\mathbf{R}_{h+1,t}$;
  end
end
2. Compute policy gradients $\frac{\partial\bar{r}_t}{\partial\boldsymbol{\vartheta}_h}$ for $t = 1, 2, \ldots, H$ as follows:
$\frac{\partial\bar{r}_1}{\partial\boldsymbol{\vartheta}_1} = (\bar{\boldsymbol{s}}_1 - \boldsymbol{\mu}_1^s)\boldsymbol{b}_1^\top$;
for $t = 2, 3, \ldots, H$ **do**
  $\frac{\partial\bar{r}_t}{\partial\boldsymbol{\vartheta}_t} = (\bar{\boldsymbol{s}}_t - \boldsymbol{\mu}_t^s)\boldsymbol{b}_t^\top$;
  for $h = t-1, t-2, \ldots, 1$ **do**
    $\frac{\partial\bar{r}_t}{\partial\boldsymbol{\vartheta}_h} = (\boldsymbol{c}_t + \boldsymbol{\vartheta}_t\boldsymbol{b}_t))(\bar{\boldsymbol{s}}_h - \boldsymbol{\mu}_h^s)^\top\boldsymbol{\delta}_h^t$;
  end
end
3. Compute the gradient of the cumulative reward:

$$\nabla_{\boldsymbol{\theta}}J(\boldsymbol{\theta};\boldsymbol{w}) = \left(\text{vec}\left(\sum_{t=1}^{H}\nabla_{\boldsymbol{\vartheta}_1}\bar{r}_t(\boldsymbol{\theta};\boldsymbol{w})\right), \ldots, \text{vec}\left(\sum_{t=1}^{H}\nabla_{\boldsymbol{\vartheta}_{H-1}}\bar{r}_t(\boldsymbol{\theta};\boldsymbol{w})\right)\right)^\top.$$

$\frac{\partial\bar{r}_{t+1}}{\partial\boldsymbol{\theta}_t}$ propagates the reward signal $\bar{r}_{t+1}$ back to $\boldsymbol{\vartheta}_t$, and the computation now uses information from the nodes r_{t+1}, $\boldsymbol{a}_{t+1}$, $\boldsymbol{s}_{t+1}$, $\boldsymbol{a}_t$ and $\boldsymbol{s}_t$ and associated edges. In Figure 3(d), we *reuse* the pathway $\mathbf{R}_{t,t} = (\boldsymbol{\beta}_t^s + \boldsymbol{\vartheta}_t\boldsymbol{\beta}_t^a)^\top\mathbf{R}_{t+1,t}$ from Figure 3(c) to compute $\mathbf{R}_{t-1,t} = (\boldsymbol{\beta}_{t-1}^s + \boldsymbol{\vartheta}_{t-1}\boldsymbol{\beta}_{t-1}^a)^\top\mathbf{R}_{t,t}$.

The formal statement of the nested backpropagation procedure is given in Algorithm 1. In Step (1), we first pass through the network to efficiently precompute the propagation pathways $\mathbf{R}_{h,t}$ through reuse, and store $\boldsymbol{\delta}_h^t$ that will be used multiple times to compute the gradients $\frac{\partial\bar{r}_t}{\partial\boldsymbol{\vartheta}_h}$ for $t = 1, 2, \ldots, H$ and $h = t-1, t-2, \ldots, 1$ in Step (2). Proposition 3 proves that this approach reduces the cost of computing $\nabla_{\boldsymbol{\theta}}J(\boldsymbol{\theta};\boldsymbol{w})$ by a factor of $\mathcal{O}(H)$ compared to a brute-force approach that does not reuse pathways. As will be shown in our computational study, the time savings can be quite significant. The proof can be found in Appendix 10.4.

Proposition 3 *Fix a model $\boldsymbol{w}$ and policy parameter vector $\boldsymbol{\theta}$. The cost to compute $\nabla_{\boldsymbol{\theta}}J(\boldsymbol{\theta};\boldsymbol{w})$ is $\mathcal{O}\left(H^2n^2m\right)$ for nested backpropagation (Algorithm 1) and $\mathcal{O}\left(H^3n^2m\right)$ for brute force.*

4.2 Policy Optimization Algorithm

Recall that the gradient of $\mathcal{J}$ is estimated using SAA in (13), with $J\left(\boldsymbol{\theta};\boldsymbol{w}^{(b)}\right)$ computed using Algorithm 1. We can now search for the optimal $\boldsymbol{\theta}$ using a recursive gradient ascent update. Given $\boldsymbol{\theta}_k$, we compute

$$\boldsymbol{\theta}_{k+1} = \Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k\nabla\widehat{\mathcal{J}}\left(\boldsymbol{\theta}_k\right)\right), \tag{16}$$

where η_k is a suitable stepsize, $\mathbb{C}$ is a closed convex feasible region as discussed in Section 3.3, and $\Pi_{\mathbb{C}}$ is a projection onto $\mathbb{C}$. We summarize some previously established properties of the projection, which will be needed for our convergence analysis later; the proofs are given in Section 2.2 of Jain and Kar (2017), and thus are omitted here. Throughout the paper, we use the usual Euclidean norm and inner product.

Definition 1 *The projection of a point $\boldsymbol{y}$ onto $\mathbb{C}$ is defined as $\Pi_{\mathbb{C}}(\boldsymbol{y}) = \arg\min_{\boldsymbol{x}\in\mathbb{C}}\frac{1}{2}\|\boldsymbol{x}-\boldsymbol{y}\|^2$.*

Proposition 4 *For any set $\mathbb{C} \subset \mathbb{R}^p$, the following inequalities hold:*

1. *If $\boldsymbol{y} \in \mathbb{C}$, then $\Pi_{\mathbb{C}}(\boldsymbol{y}) = \boldsymbol{y}$.*
2. *For any $\boldsymbol{y} \in \mathbb{R}^p$ and $\boldsymbol{x} \in \mathbb{C}$, $\|\Pi_{\mathbb{C}}(\boldsymbol{y}) - \boldsymbol{y}\| \le \|\boldsymbol{x} - \boldsymbol{y}\|$.*

Algorithm 2: DBN-RL algorithm for policy optimization.

Input: the maximum number of episodes K; the convex constraint set $\mathbb{C}$; initial parameter $\boldsymbol{\theta}_0$ for linear policy $\boldsymbol{\pi}_{\boldsymbol{\theta}_0}(\boldsymbol{s})$, $\forall \boldsymbol{s} \in \mathcal{S}, \boldsymbol{\theta} \in \mathbb{C}$; posterior distribution $p(\boldsymbol{w}|\mathcal{D})$ for process knowledge graph model;

for $k = 1, 2, \ldots, K$ **do**

1. Generate B posterior samples of process model parameters $\boldsymbol{w}_k^{(b)} \sim p(\boldsymbol{w}|\mathcal{D})$ with $b = 1, 2, \ldots, B$;

for $b = 1, 2, \ldots, B$ **do**

if $\boldsymbol{w}_k^{(b)} \in \Omega$ **then**

Calculate the policy gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k; \boldsymbol{w}_k^{(b)})$ by using Algorithm 1;

else

$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k; \boldsymbol{w}_k^{(b)}) = 0$;

end

end

3. Calculate $\nabla \widehat{\mathcal{J}}(\boldsymbol{\pi}_{\boldsymbol{\theta}}) = \frac{1}{B} \sum_{b=1}^{B} \nabla_{\boldsymbol{\theta}} J\left(\boldsymbol{\pi}_{\boldsymbol{\theta}}; \boldsymbol{w}_k^{(b)}\right)$;

4. Obtain the gradient mapping $\widehat{g}_c(\boldsymbol{\theta}_k) = \frac{1}{\eta_k}(\tilde{\boldsymbol{\theta}}_k - \boldsymbol{\theta}_k)$, where $\tilde{\boldsymbol{\theta}}_k = \Pi_C\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right)$;

5. Update the policy parameters $\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \eta_k \widehat{g}_c(\boldsymbol{\theta}_k)$.

end

3. *If $\mathbb{C}$ is convex, $\Pi_{\mathbb{C}}$ is non-expansive. That is, for any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^p$,*
$$\|\Pi_{\mathbb{C}}(\boldsymbol{x}) - \Pi_{\mathbb{C}}(\boldsymbol{y})\|^2 \leq \langle \Pi_{\mathbb{C}}(\boldsymbol{x}) - \Pi_{\mathbb{C}}(\boldsymbol{y}), \boldsymbol{x} - \boldsymbol{y} \rangle \leq \|\boldsymbol{x} - \boldsymbol{y}\|^2.$$

4. *For any convex set $\mathbb{C} \in \mathbb{R}^p$, $\boldsymbol{y} \in \mathbb{R}^p$ and $\boldsymbol{x} \in \mathbb{C}$, we have $\langle \boldsymbol{x} - \Pi_{\mathbb{C}}(\boldsymbol{y}), \boldsymbol{y} - \Pi_{\mathbb{C}}(\boldsymbol{y}) \rangle \leq 0$.*

It is sometimes convenient to rewrite (16) as

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \eta_k \widehat{g}_k(\boldsymbol{\theta}_k) \quad \text{with} \quad \widehat{g}_k(\boldsymbol{\theta}) = \frac{1}{\eta_k}\left(\Pi_{\mathbb{C}}\left(\boldsymbol{\theta} + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta})\right) - \boldsymbol{\theta}\right) \tag{17}$$

where $\widehat{g}_k(\boldsymbol{\theta})$ can be viewed as a generalized gradient estimator. This representation is identical to (16), except that (17) resembles a traditional unconstrained gradient ascent update.

The complete statement of the policy optimization procedure, which we call DBN-RL ("DBN-assisted reinforcement learning"), is given in Algorithm 2. In each iteration, we generate B model parameters, $\left\{\boldsymbol{w}^{(b)}\right\}_{b=1}^{B}$, by sampling from the posterior distribution $p(\boldsymbol{w}|\mathcal{D})$, then use nested backpropagation to calculate gradients for each individual model. We then average over the models and apply (17) to update the policy parameters. It is worth noting that the only randomness in this stochastic gradient method comes from the use of SAA to approximate the expectation in (9) with an average over B posterior samples in (13). The computational results of Theorem 1 and Algorithm 1 allow us to explicitly compute $\nabla_{\boldsymbol{\theta}} J\left(\boldsymbol{\theta}; \boldsymbol{w}^{(b)}\right)$ for $b = 1, 2, \ldots, B$.

5 Convergence Analysis

In this section, we prove the convergence of DBN-RL to a local optimum of $\mathcal{J}$ and characterize the convergence rate. Recall from Proposition 2 that $\mathcal{J}$ is non-convex in $\boldsymbol{\theta}$. Consequently, only local convergence can be guaranteed, as is typical of stochastic gradient ascent methods.

There is a rich literature on the convergence analysis of this class of algorithms, covering a wide variety of settings. However, the proofs often rely on very subtle nuances in model assumptions, e.g., on the level of noise or the smoothness of the objective function. The classical theory (see, e.g., Kushner and Yin 2003) has focused on almost sure convergence (e.g., of $\nabla \mathcal{J}(\boldsymbol{\theta}_k)$ to zero). The work by Bertsekas and Tsitsiklis (2000) gives some of the weakest assumptions for this type of convergence when the objective is non-convex.

Nemirovski et al. (2009) pioneered a different style of analysis which derived convergence rates. This initial work required weakly convex objectives, but the non-convex setting was variously studied by Ghadimi and Lan (2013), Jain and Kar (2017), and Li and Orabona (2019). Our analysis belongs to this overall stream, but derives convergence rates in the presence of a projection operator, with weaker assumptions than past papers studying similar settings.

We first note that the specific objective function optimized by DBN-RL was shown to be L-smooth in Lemma 1. A direct consequence of this property (Lemma 1.2.3 of Nesterov 2003), used in our analysis, is that

$$|\mathcal{J}(\boldsymbol{y}) - \mathcal{J}(\boldsymbol{x}) - \langle \nabla \mathcal{J}(\boldsymbol{x}), \boldsymbol{y} - \boldsymbol{x} \rangle| \leq L\|\boldsymbol{x} - \boldsymbol{y}\|^2. \tag{18}$$

For notational simplicity, we use $\mathbb{E}$ to refer to the expectation over the posterior distribution $p(\boldsymbol{w}|\mathcal{D})$, since this distribution is the only source of randomness in the algorithm. We also let

$$A_k = \left\{\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) \in \mathbb{C}\right\} \tag{19}$$

be the event that the updated gradient remains in the feasible region (i.e., we do not need to project it back onto $\mathbb{C}$) after the kth episode.

Three additional assumptions are required for the convergence proof.

Assumption 1 (Boundary condition) *For any $\boldsymbol{\theta} \in \partial\mathbb{C}$ and model $\boldsymbol{w} \in \mathcal{W}$, there exists a constant $c_0 > 0$ such that, when the stepsize $\eta \leq c_0$, we have*

$$\boldsymbol{\theta} + \eta \frac{\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}; \boldsymbol{w})}{\|\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}; \boldsymbol{w})\|} \in \mathbb{C}.$$

Assumption 1 suggests that, at any point on the boundary of the feasible region, the gradient always points toward the interior of $\mathbb{C}$. For example, one might consider a situation where the feasible region is large enough to include all local maxima. This assumption is reasonable for biomanufacturing, because in such an application $\mathbb{C}$ should be defined based on FDA regulatory requirements; any batch outside the required region will induce a massive penalty.

Assumption 2 (Noise level) *For any $\boldsymbol{\theta} \in \mathbb{C}$, we have $\mathbb{E}[\|\nabla \widehat{\mathcal{J}}(\theta) - \nabla \mathcal{J}(\theta)\|^4]^{1/4} \leq \sigma$.*

Classical stochastic approximation theory (Kushner and Yin 2003) assumes uniformly bounded variance, i.e., $\mathbb{E}\left[\|\nabla \widehat{\mathcal{J}}(\theta) - \nabla \mathcal{J}(\theta)\|^2\right] \leq \sigma^2$. This assumption is weaker than Assumption 2, but the results will also be weaker since the classical theory is primarily concerned with almost sure convergence only. If we compare against existing work on convergence *rates*, Assumption 2 is *weaker* than other widely used assumptions such as $\mathbb{E}[\exp(\|\nabla \widehat{\mathcal{J}}(\theta) - \nabla \mathcal{J}(\theta)\|^2/\sigma^2)] \leq \exp(1)$, used in Nemirovski et al. (2009) and Li and Orabona (2019). For non-convex and strongly convex cases, one can also find other assumptions, such as boundedness $\|\nabla \widehat{\mathcal{J}}(\theta) - \nabla \mathcal{J}(\theta)\|^2 \leq S$ of the stochastic gradient itself (Li and Orabona 2019), or boundedness $\mathbb{E}\left[\|\nabla \widehat{\mathcal{J}}(\theta)\|^2\right] \leq \sigma^2$ of the second moment of the gradient estimator (Shalev-Shwartz et al. 2011, Niu et al. 2011). Assumption 2 is weaker than all of these.

Assumption 3 *The global maximum $\boldsymbol{\theta}^\star$ lies in the interior of $\mathbb{C}$, that is, $\boldsymbol{\theta}^\star \in \mathbb{C} \backslash \partial\mathbb{C}$ and $\mathcal{J}(\boldsymbol{\theta}) \leq \mathcal{J}(\boldsymbol{\theta}^\star)$ for any $\boldsymbol{\theta} \in \mathbb{C}$.*

Assumption 3 guarantees that the feasible region is large enough to include the optimal solution, complementing Assumption 1, which ensures that the gradient points us back toward the interior if we are ever close to leaving the feasible region.

We can now proceed with the analysis; proofs of the main result and supporting lemmas can be found in Appendix 10.6-10.10. First, by applying Lemma 1 and Assumption 3, we can show that the true policy gradient $\nabla \mathcal{J}(\boldsymbol{\theta})$ (not the estimator of this quantity!) is bounded on the feasible region.

Corollary 1 *For any $\boldsymbol{x} \in \mathbb{C}$, we have $\|\nabla \mathcal{J}(\boldsymbol{x})\| \leq \max_{\boldsymbol{x} \in \mathbb{C}} \|\nabla \mathcal{J}(\boldsymbol{x})\| \leq G$, where $G = L \cdot \max_{\boldsymbol{y} \in \mathbb{C}} \|\boldsymbol{y} - \boldsymbol{\theta}^\star\|^2$.*

Recalling the definition of A_k in (19), we let $p(A_k)$ be the probability that the updated gradient after the kth iteration remains in $\mathbb{C}$. We then show that, on almost every sample path, this will happen for all sufficiently large k.

Lemma 2 *Let Assumptions 1-3 hold and suppose $\sum_{k=1}^{\infty} \eta_k^2 < \infty$. Then,*

$$\lim_{k \to \infty} p(A_k) = \lim_{k \to \infty} P\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) \in \mathbb{C}\right) = 1.$$

The final convergence result connects $p(A_k)$ to the averaged expected norm $\frac{1}{K}\sum_{k=1}^{K} \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right]$ of the true policy gradient.

Theorem 2 *Let Assumptions 1-3 hold, and suppose that the stepsize satisfies $\sum_{k=1}^{\infty} \eta_k^2 < \infty$. Then,*

$$\frac{1}{K}\sum_{k=1}^{K} \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] \leq \frac{\frac{2}{\eta_1}(\mathcal{J}^\star - \mathcal{J}(\boldsymbol{\theta}_1)) + 2(G^2 + G\sigma)\sum_{k=1}^{K}(1 - p(A_k))^{1/2} + 8L\sigma^2 \sum_{k=1}^{K} \eta_k}{K}.$$

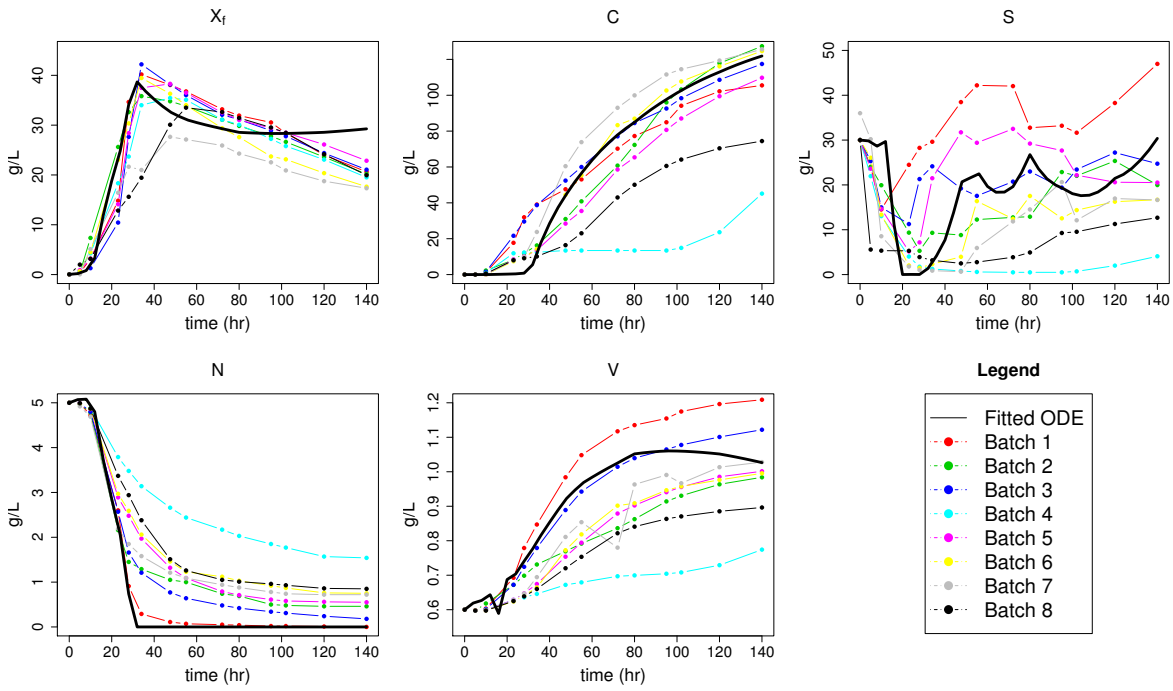


Figure 4: ODE trajectory (in black) and 8 batches of real lab experiment trajectories. Batches 4 and 8, which have low productivity, are two experiments with low oil feed.

Thus, the convergence rate of our algorithm is connected to the behavior of $p\left(A_k\right)$. Combining Lemma 2 with Theorem 2, it straightforwardly follows that

$$\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\left[\|\nabla\mathcal{J}(\boldsymbol{\theta}_k)\|^2\right]\to 0.$$

6 Empirical Study

We present a case application of our approach to multi-phase fermentation of *Yarrowia lipolytica*, a process in which viable cells grow and produce the biological substance of interest. Such operations drive productivity and typically contribute $70-80\%$ of the total production cost (Harrison et al. 2015, Straathof 2011). Based on real lab experiment data and existing domain knowledge on bioprocess mechanisms, in Section 6.1, we first develop a simulator and use it to generate R runs of fermentation process training data set, denoted by $\mathcal{D}$, to assess the performance of proposed DBN-RL and compare it with a state-of-art RL candidate approach. Section 6.2 presents the main empirical results, with some additional issues investigated in Section 6.3; finally, Section 6.4 discusses the interpretability of the results.

6.1 Problem Context and Model

Based on existing domain knowledge on fermentation process kinetics and real 8 batches of lab experimental data, we create a simulator representing the underlying "true" stochastic process. We first fit an ODE-based nonlinear kinetics model and use Wiener process to model the intrinsic stochastic uncertainty; see more details in Appendix 9. Then, we generate the fermentation process *training* data set $\mathcal{D}$ from this stochastic mechanistic simulation model with actions chosen by using ϵ-greedy method.

We have a five-dimensional continuous state variable $\boldsymbol{s}=(X_f,C,S,N,V)$, where X_f represents lipid-free cell mass; C measures citrate, the actual "product" to be harvested at the end of the bioprocess, generated by the cells' metabolism; S and N are amounts of substrate (a type of oil) and nitrogen, both used for cell growth and production; and V is the working volume of the entire batch. We also have one scalar continuous CPP action $a=F_S$, which represents the feed rate, or the amount of new substrate given to the cell in one unit of time.

Using 8 batches of real data from fed-batch fermentation experiments, we first calibrate a nonlinear kinetic model for the process dynamics, using least squares to fit parameter values. The details of this model and estimation are given in Appendix 9. Figure 4 shows the trajectories of the five dimensions of the state variable under each of the 8 real batches as well as under the estimated kinetic model. One can see that there is a great deal of variation between real experiments; the given data are not sufficient to estimate intrinsic stochastic uncertainty because each experiment was conducted with very different decision parameters.

We chose to set the inherent stochastic uncertainty of state $\boldsymbol{s}_t = (X_{ft}, C_t, S_t, N_t, V_t)$ at time t as

$$\sigma(s_t) = \frac{1}{8\kappa} \sum_{i=1}^{8} s_t^{(i)}, \tag{20}$$

a multiple of the estimated mean over 8 experiments. In our experiments, we considered $\kappa \in \{10, 25, \infty\}$, reflecting different levels of process uncertainty. The stochastic uncertainty can change with time and also cross different state variables. We incorporate $\sigma(s_t)$ as variation parameter into the Wiener process; see the stochastic simulation model in Appendix 9.

The initial state $\boldsymbol{s}_1$ was chosen randomly to account for raw material variation. Our process knowledge graph consists of 215 nodes and 770 edges with 36 time measurement steps (proceeding in increments of four hours, up to the harvest time of 140 hours). There are 35 transitions, 5 CQA nodes (one for each continuous dimension) and one CPP node per time period, leading to a total of 985 parameters to estimate using Gibbs sampling. The region $\mathcal{W}$ of valid model parameters was chosen to ensure that each parameter is bounded by a large constant 10^{10}.

Then, the simulator is used to generate R experiment runs of training dataset $\mathcal{D}$. In these experimental data generation, we chose the feed rate action according to the ϵ-greedy to balance exploration and exploitation (Sutton and Barto 2018):

$$a_t = \begin{cases} a_t^h + \mathcal{N}(0, \bar{a}_t/10), & \text{with probability } 0.7, \\ \text{Unif}(0, \max_{t \in \mathcal{T}}\{\bar{a}_t\}), & \text{with probability } 0.3 \end{cases}$$

where $\bar{a}_t$ denotes the maximum feed rate across our 8 real experiments at time t, and a_t^h denotes the average feed rate across these experiments. In words, we randomly choose either the average feed rate from the real data (with some noise), or a uniformly distributed quantity between zero and the maximum feed rate.

The reward function at time t was formulated by consultation with bioprocess experts, and is determined by the citrate concentration C_t and feed rate a_t according to

$$r(\boldsymbol{s}_t, a_t) = \begin{cases} -15 + 1.29 C_t, & \text{if } t = H, \\ -534.52 a_t, & \text{if } 0 \le t < H. \end{cases}$$

This structure reflects the fact that, during the process, the operating cost is primarily driven by the substrate, and the main source of reward is the product revenue collected at the harvest time.

The linear policy (10) is non-stationary; since there are 35 state transitions, five state nodes and one action node, the total number of policy parameters is 175. Based on consultation with bioprocess experts, the feasible region $\mathbb{C}$ was chosen by constraining each individual parameter to an interval. The lower and upper bounds are the same for all parameters corresponding to a particular CQA (dimension of the state variable). For cell mass and citrate, we used the interval $[0, 0.3]$ as the action (feed rate of substrate) always positively affects cell growth and citrate production. For substrate, we used the interval $[-0.1, 0.1]$ because excessively high substrate concentration may have adverse effects. For nitrogen, we used the interval $[-0.1, 0.02]$, and for volume, we used $[-0.7, 0.5]$.

6.2 Performance Comparison

We compare DBN-RL against Deep Deterministic Policy Gradient (DDPG), a state-of-the-art model-free RL algorithm designed for continuous control (Lillicrap et al. 2015), as well as with a "human" policy inferred from the 8 real experiments (see Figure 4). Although we are not given an exact policy used by human experts (and it is unlikely that they follow any explicit policy in practice), their observed actions are based on their domain knowledge and so the inferred policy may be quite competitive in some situations. Both DBN-RL and DDPG use the same set of training data. For the human expert policy, we only consider $R = 8$ as these are all the human experts we have.

Table 1 reports the total reward, as well as the final citrate titer (amount of product harvested at the end of 140 hours), for each policy. A total of 30 macroreplications was conducted; this number was sufficient to obtain statistically significant performance estimates. We see that, under the smallest sample size $R = 8$, the human expert policy achieves the best performance, as might be expected from domain experts. However, with only a few additional samples from the kinetic model ($R = 15$), DBN-RL can improve on the human policy, with additional improvement as R increases.

Table 1: Reward and final citrate titer of DBN-RL, DDPG and human policy (30 macro-replications).

Algorithms		DBN-RL				DDPG				Human			
Sample Size	κ	Reward Mean	Reward SE	Titer (g/L) Mean	Titer (g/L) SE	Reward Mean	Reward SE	Titer (g/L) Mean	Titer (g/L) SE	Reward Mean	Reward SE	Titer (g/L) Mean	Titer (g/L) SE
$R = 8$	10	103.44	2.50	101.15	1.11	-		-		113.47	4.26	102.12	3.88
	25	108.13	2.06	103.15	1.11	-		-		116.19	4.03	103.71	3.45
	∞	114.96	1.73	105.93	0.94	-		-		118.76	3.56	104.86	3.10
$R = 15$	10	119.75	1.60	110.62	0.84	17.65	11.53	39.16	8.27	-		-	
	25	120.11	1.64	111.62	0.93	23.35	10.83	47.11	8.40	-		-	
	∞	121.35	1.75	111.98	0.90	20.65	10.24	49.04	7.12	-		-	
$R = 50$	10	126.94	2.15	115.29	1.19	24.65	9.00	50.33	6.41	-		-	
	25	128.34	1.23	115.13	4.52	20.44	8.04	45.93	6.10	-		-	
	∞	131.97	0.47	116.05	0.41	21.65	7.18	49.33	5.45	-		-	
$R = 100$	10	128.90	1.25	115.89	0.60	31.36	10.98	45.21	8.24	-		-	
	25	130.73	0.95	116.02	0.59	32.89	10.14	50.23	8.60	-		-	
	∞	131.92	0.73	116.35	0.45	35.89	9.88	51.81	7.85	-		-	
$R = 400$	10	130.40	0.57	116.45	0.38	37.23	9.12	50.43	8.28	-		-	
	25	131.01	0.37	116.75	0.30	37.43	9.95	52.43	8.87	-		-	
	∞	133.04	0.12	118.12	0.14	39.77	9.23	55.16	9.29	-		-	
$R = 3000$	∞	-		-		119.28	0.84	104.78	0.74	-		-	

The DDPG policy is much less efficient than DBN-RL, because it is entirely model-free and cannot benefit from known problem structure. For demonstration purposes, we report the performance of DDPG for a very large sample size of $R = 3000$, simply to show that this policy is indeed able to do reasonably well (at least outperforming the human policy) when given enough data. However, even these numbers are not as good as what DBN-RL can achieve with just $R = 15$ samples. In this we see a major advantage of our approach, namely, the ability to use problem structure from kinetic models to learn much more efficiently. The results show that DBN-RL is a robust and sample-efficient algorithm that can achieve human level control given very small amounts of real experimental data.

6.3 Model Risk and Computational Efficiency

We present additional numerical results to explore two important aspects of DBN-RL, namely the benefits of accommodating model risk, and the computational savings obtained by using the nested backpropagation procedure (Algorithm 1).

First, we compare DBN-RL with posterior sampling (as in Algorithm 2) to a version where the posterior samples $\boldsymbol{w}^{(b)}$ are replaced by a single point estimate (i.e., the posterior mean) of the model parameters. This second version ignores model risk. Table 2 shows that accounting for model risk can produce a better policy when the amount of data is small. This can arise in biomanufacturing when a reliable kinetic model is not available.

Table 2: Performance of DBN-RL with and without model risk ($\kappa = 10$, 30 macro-replications).

Sample size	Metrics	DBN-RL with MR	DBN-RL ignoring MR	p-value
$R = 8$	Reward	103.44 (2.50)	95.39 (2.19)	0.02
	Titer (g/L)	101.15 (1.11)	92.64 (0.96)	<0.001
$R = 15$	Reward	119.75 (1.60)	118.35 (1.81)	0.56
	Titer (g/L)	118.35 (0.84)	109.12 (0.94)	0.24
$R = 50$	Reward	126.94 (2.15)	129.16 (1.37)	0.39
	Titer (g/L)	115.29 (1.19)	115.08 (0.60)	0.88

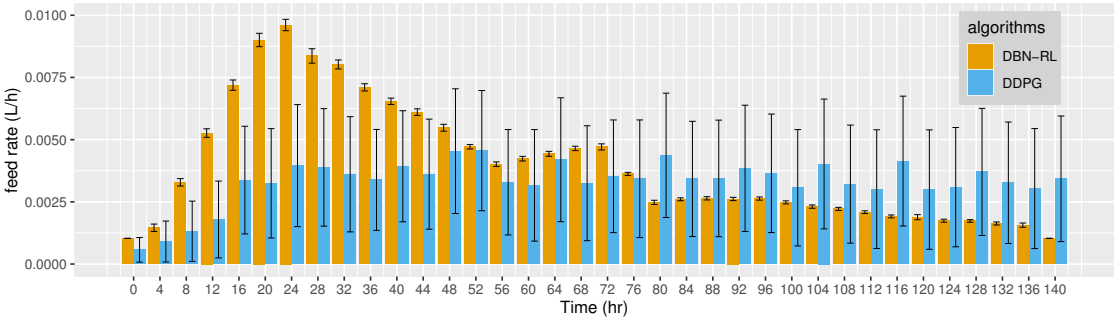
Next, we compare the computational cost of DBN-RL with and without nested backpropagation. When Algorithm 1 is not used, the policy gradient is computed using brute force. Table 3 reports mean computational time in minutes (averaged over 30 macro-replications) for different sample sizes R and time horizons H. For reference, we also report the training time for the Bayesian network (i.e., the Bayesian inference calculations, which are unrelated to policy optimization). It is clear that the brute-force method scales very poorly with the process complexity H. In practice, H tends to be much greater than the sample size R, meaning that nested backpropagation is much more scalable.

Table 3: Computational time (in minutes) of DBN-RL with and without nested backpropagation.

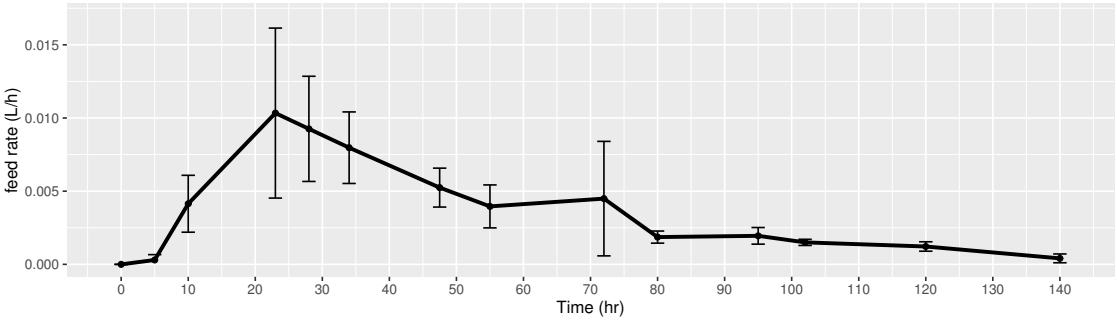
Horizon	$R = 15$			$R = 100$			$R = 400$		
	Training	NBP	Brute Force	Training	NBP	Brute Force	Training	NBP	Brute Force
$H = 8$	1.1 (0.1)	0.8 (0.1)	5.9 (0.2)	3.9 (0.1)	0.8 (0.1)	5.6 (0.2)	14.9 (0.3)	0.8 (0.1)	6.1 (0.2)
$H = 15$	2.4 (0.7)	2.1 (0.4)	27.4 (1.2)	8.7 (0.3)	2.4 (0.3)	26.9 (1.2)	31.5 (2.3)	2.0 (0.4)	28.1 (1.1)
$H = 36$	4.1 (0.2)	9.1 (0.3)	302.3 (5.3)	17.9 (0.1)	9.7 (0.8)	312.3 (5.6)	59.7 (0.6)	10.1 (0.7)	310.5 (6.1)

6.4 Interpretability of DBN-RL Policies

A well-known limitation of model-free reinforcement learning techniques (such as those based on deep networks) is their lack of interpretability. This is a serious concern in biomanufacturing, where a single batch of product can easily be worth over $\$1M$. Human experts are closely involved in practical implementation, and require explanations of the features used in a model or the decisions made by a policy. These requirements can be met by our approach. First, the network structure provides an intuitive visualization of the quantitative associations between CPPs and CQAs. Second, the policy learned from DBN-RL behaves in a way that is understandable to human experts, as demonstrated by the following example.



(a) Feeding profiles of DBN-RL and DDPG with 95% confidence intervals. (R= 400)



(b) Feeding profile from human experts: averaged trajectory with 95% confidence intervals.

Figure 5: Feeding profiles obtained from (a) RL algorithms and (b) human experts.

We fix an initial state $\boldsymbol{s}_1 = (0.05, 0, 30, 5, 0.6)$ and simulate the actions (feeding rates) taken over time by following the policies obtained from DBN-RL, DDPG, and human experts. For each time t, we then average the action a_t and plot these averages over time to obtain a *feeding profile* for each type of policy. These profiles are shown in Figure 5. The policies learned by DBN-RL suggest starting out with low initial feeding rates, because the high initial substrate ($S_1 = 30$ g/L) is sufficient for biomass development (providing enough carbon sources for cell proliferation and early citrate formation). DBN-RL then ramps up feeding rapidly until the amount of substrate reaches approximately 10 g/L, when the biomass growth rate is highest. After 24 hours, DBN-RL begins to reduce the feeding rate, and generally continues doing so until the end of the process. This occurs because the life cycle of cells moves from the growth phase to the production phase at around 24 hours, and less substrate is required to support production. Comparing with Figure 5b, we find that the feeding profile of DBN-RL has a very similar curve to that of the human expert policy, and even peaks around the same time. On the other hand, DDPG does not supply enough substrate for biomass growth during the

first 30 hours, lowering the cell growth rate; it then attempts to compensate later on, but this is ineffective because of intensive inhibition from high substrate concentration.

7 Conclusion

We have presented new models and algorithms for optimization of control policies on a Bayesian knowledge graph. This approach is especially effective in engineering problems with complex structure derived from physics models. Such structure can be partially extracted and turned into a prior for the Bayesian network. Furthermore, mechanistic models can also be used to provide additional information, which becomes very valuable in the presence of highly complex nonlinear dynamics with very small amounts of available pre-existing data. All of these issues arise in the domain of biomanufacturing, and we have demonstrated that our approach can achieve human-level control using as few as 8 lab experiments, while state-of-the-art model-free methods struggle due to their inability to incorporate known structure in the process dynamics.

As synthetic biology and industrial biotechnology continue to adopt more complex processes for the generation of new drug products, from monoclonal antibodies (mAbs) to cell/gene therapies, data-driven and model-based control will become increasingly important. This work presents compelling evidence that model-based reinforcement learning can provide competitive performance and interpretability in the control of these important systems.

References

R. A. Rader. FDA biopharmaceutical product approvals and trends in 2012. *BioProcess International*, 11(3):18–27, 2013.

L. Zhou, N. Xu, Y. Sun, and X. M. Liu. Targeted biopharmaceuticals for cancer treatment. *Cancer Letters*, 352(2):145–151, 2014.

T. T. Le, J. P. Cramer, R. Chen, and S. Mayhew. Evolution of the COVID-19 vaccine development landscape. *Nature Reviews Drug Discovery*, 19(10):667–668, 2020.

R. Cintron. *Human Factors Analysis and Classification System Interrater Reliability for Biopharmaceutical Manufacturing Investigations*. PhD thesis, Walden University, 2015.

Moo Sun Hong, Kristen A Severson, Mo Jiang, Amos E Lu, J Christopher Love, and Richard D Braatz. Challenges and opportunities in biopharmaceutical manufacturing control. *Computers & Chemical Engineering*, 110:106–114, 2018.

Sarantos Kyriakopoulos, Kok Siong Ang, Meiyappan Lakshmanan, Zhuangrong Huang, Seongkyu Yoon, Rudiyanto Gunawan, and Dong-Yup Lee. Kinetic modeling of mammalian cell culture bioprocessing: The quest to advance biomanufacturing. *Biotechnology Journal*, 13(3):1700229, 2018.

Amos E. Lu, Joel A. Paulson, Nicholas J. Mozdzierz, Alan Stockdale, Ashlee N. Ford Versypt, Kerry R. Love, J. Christopher Love, and Richard D. Braatz. Control systems technology in the advanced manufacturing of biologic drugs. In *Proceedings of the IEEE Conference on Control Applications*, pages 1505–1515, 2015.

Catalina Valencia Peroni, Niket S Kaisare, and Jay H Lee. Optimal control of a fed-batch bioreactor using simulation-based approximate dynamic programming. *IEEE Transactions on Control Systems Technology*, 13(5):786–790, 2005.

Chongyang Liu, Zhaohua Gong, Bangyu Shen, and Enmin Feng. Modelling and optimal control for a fed-batch fermentation process. *Applied Mathematical Modelling*, 37(3):695–706, 2013.

Tugce Martagan, Ananth Krishnamurthy, Peter A. Leland, and Christos T. Maravelias. Performance guarantees and optimal purification decisions for engineered proteins. *Operations Research*, 66(1):18–41, 2017.

Tugce Martagan, A Krishnamurthy, and P. Leland. Managing trade-offs in protein manufacturing: How much to waste? *Manufacturing & Service Operations Management*, 2019a. doi:https://doi.org/10.1287/msom.2018.0740. Published online on April 24.

Tugce Martagan, Yasemin Limon, Ananth Krishnamurthy, Tom Foti, and Peter Leland. Aldevron accelerates growth using operations research in biomanufacturing. *INFORMS Journal on Applied Analytics*, 49(2):137–153, 2019b.

S. P. K. Spielberg, R. B. Gopaluni, and P. D. Loewen. Deep reinforcement learning approaches for process control. In *2017 6th International Symposium on Advanced Control of Industrial Processes (AdCONIP)*, pages 201–206, 2017. doi:10.1109/ADCONIP.2017.7983780.

Steven Spielberg, Aditya Tulsyan, Nathan P Lawrence, Philip D Loewen, and R Bhushan Gopaluni. Deep reinforcement learning for process control: A primer for beginners. *arXiv preprint arXiv:2004.05490*, 2020.

Neythen J Treloar, Alex JH Fedorec, Brian Ingalls, and Chris P Barnes. Deep reinforcement learning for the control of microbial co-cultures in bioreactors. *PLoS computational biology*, 16(4):e1007783, 2020.

A. V. Bankar, A. R. Kumar, and S. S. Zinjarde. Environmental and industrial applications of Yarrowia lipolytica. *Applied Microbiology and Biotechnology*, 84(5):847–865, 2009.

Carl-Fredrik Mandenius, Nigel J Titchener-Hooker, et al. *Measurement, monitoring, modelling and control of bioprocesses*, volume 132. Springer, 2013.

Anurag S. Rathore, Nitish Bhushan, and Sandip Hadpe. Chemometrics applications in biotech processes: A review. *Biotechnology Progress*, 27(2):307–315, 2011. doi:https://doi.org/10.1002/btpr.561. URL `https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/btpr.561`.

A. P. Teixeira, N. Carinhas, J. M. L. Dias, P. Cruz, P. M. Alves, M. J. T. Carrondo, and R. Oliveira. Hybrid semi-parametric mathematical systems: Bridging the gap between systems biology and process engineering. *Journal of Biotechnology*, 132(4): 418–425, 2007.

Jon C Gunther, Jeremy S Conner, and Dale E Seborg. Process monitoring and quality variable prediction utilizing pls in industrial fed-batch cell culture. *Journal of Process Control*, 19(5):914–921, 2009.

Mo Jiang and RD Braatz. Integrated control of continuous (bio) pharmaceutical manufacturing. *American Pharmaceutical Review*, 19(6):110–115, 2016.

Richard Lakerveld, Brahim Benyahia, Richard D Braatz, and Paul I Barton. Model-based design of a plant-wide control strategy for a continuous pharmaceutical plant. *AIChE Journal*, 59(10):3671–3685, 2013.

Tugce Martagan, Ananth Krishnamurthy, and Christos T. Maravelias. Optimal condition-based harvesting policies for biomanufacturing operations with failure risks. *IIE Transactions*, 48(5):440–461, 2016.

Hua Zheng, Ilya O. Ryzhov, Wei Xie, and Judy Zhong. Personalized multimorbidity management for patients with type 2 diabetes using reinforcement learning of electronic health records. *Drugs*, Feb 2021. ISSN 1179-1950. doi:10.1007/s40265-020-01435-4. URL `https://doi.org/10.1007/s40265-020-01435-4`.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Wei Xie, Bo Wang, Cheng Li, Jared Auclair, and Peter Baker. Bayesian network based risk and sensitivity analysis for production process stability control. *Preprint, submitted September*, 10:2019, 2020.

A. E. Gelfand. Gibbs sampling. *Journal of the American Statistical Association*, 95(452):1300–1304, 2000.

W. B. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality (2nd ed.)*. John Wiley and Sons, New York, 2011.

A. T. C. Goh. Back-propagation neural networks for modeling complex systems. *Artificial Intelligence in Engineering*, 9(3):143–151, 1995.

Prateek Jain and Purushottam Kar. Non-convex optimization for machine learning. *arXiv preprint arXiv:1712.07897*, 2017.

H. Kushner and G. Yin. *Stochastic approximation and recursive algorithms and applications (2nd ed.)*, volume 35. Springer Science & Business Media, 2003.

Dimitri P Bertsekas and John N Tsitsiklis. Gradient convergence in gradient methods with errors. *SIAM Journal on Optimization*, 10 (3):627–642, 2000.

Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609, 2009.

Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

Xiaoyu Li and Francesco Orabona. On the convergence of stochastic gradient descent with adaptive stepsizes. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 983–992. PMLR, 2019.

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2003.

Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, 127(1):3–30, 2011.

Feng Niu, Benjamin Recht, Christopher Ré, and Stephen J Wright. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. *arXiv preprint arXiv:1106.5730*, 2011.

Roger G Harrison, Paul W Todd, Scott R Rudge, and Demetri P Petrides. Bioprocess design and economics. In *Bioseparations Science and Engineering*. Oxford University Press, 2015.

AJJ Straathof. The proportion of downstream costs in fermentative production processes. 2011.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Appendices

Section 8 briefly describes key computations used in the Gibbs sampling procedure. Section 9 provides additional domain information about the application studied in Section 6. Specifically, we give the complete details of the kinetic models used in the construction and estimation of the Bayesian network. Section 10 gives proofs of all results stated in the main text.

8 Details of Gibbs Sampling Procedure

Given the data $\mathcal{D}$, the posterior distribution of $\boldsymbol{w}$ is proportional to $p(\boldsymbol{w}) \prod_{n=1}^{R} p\left(\boldsymbol{\tau}^{(n)}|\boldsymbol{w}\right)$. The Gibbs sampling technique can be used to sample from this distribution. Overall, the procedure is quite similar to the one laid out in Xie et al. (2020), so we will only give a brief description of the computations used in our setting.

For each node X in the network, let $Ch(X)$ be the set of child nodes (direct successors) of X. The full likelihood $p(\mathcal{D}|\boldsymbol{w})$ becomes

$$p(\mathcal{D}|\boldsymbol{w}) = \prod_{i=1}^{R} p(\boldsymbol{\tau}_i|\boldsymbol{w}) = \prod_{i=1}^{R}\prod_{t=1}^{H}\left[\prod_{k=1}^{m}\mathcal{N}\left(\lambda_t^k, (\sigma_t^k)^2\right)\prod_{k=1}^{n}\mathcal{N}\left(\mu_{t+1}^k + \sum_{X_t^j \in Pa(s_{t+1}^k)} \beta_t^{jk}(X_t^j - \mu_t^j), (v_{t+1}^k)^2\right)\right],$$

For the model parameters $\boldsymbol{\mu}^a, \boldsymbol{\mu}^s, \boldsymbol{\sigma}^2\boldsymbol{v}^2, \boldsymbol{\beta}$, we have the conjugate prior

$$p(\boldsymbol{\mu}^a, \boldsymbol{\mu}^s, \boldsymbol{\sigma}^2\boldsymbol{v}^2, \boldsymbol{\beta}) = \prod_{t=1}^{H}\left(\prod_{k=1}^{m} p(\lambda_t^k)p((\sigma_k^t)^2)\prod_{k=1}^{n} p(\mu_t^k)p((v_t^k)^2)\prod_{i\neq j} p(\beta_t^{ij})\right),$$

where

$$\begin{aligned}
p(\lambda_t^k) &= \mathcal{N}\left(\lambda_t^{k(0)}, \left(\delta_{t,k}^{(\lambda)}\right)^2\right), \quad p(\mu_t^k) = \mathcal{N}\left(\mu_t^{k(0)}, \left(\delta_{t,k}^{(\mu)}\right)^2\right), \quad p(\beta_t^{ij}) = \mathcal{N}\left(\beta_t^{ij(0)}, \left(\delta_{t,ij}^{(\beta)}\right)^2\right) \\
p\left((\sigma_t^k)^2\right) &= \text{Inv-}\Gamma\left(\frac{\kappa_{t,k}^{(\sigma)}}{2}, \frac{\rho_{t,k}^{(\sigma)}}{2}\right), \quad p((v_t^k)^2) = \text{Inv-}\Gamma\left(\frac{\kappa_{t,k}^{(v)}}{2}, \frac{\rho_{t,k}^{(v)}}{2}\right),
\end{aligned}$$

where Inv-Γ denotes the inverse-gamma distribution.

We now state the posterior conditional distribution for each model parameter; the detailed derivations are omitted since they proceed very similarly to those in Xie et al. (2020). Let $\boldsymbol{\mu}^a_{-t,k}, \boldsymbol{\mu}^s_{-t,k}, \boldsymbol{\sigma}^2_{-t,k}, \boldsymbol{v}^2_{-t,k}$ and $\boldsymbol{\beta}_{-t,jk}$ denote the collection of parameters $\boldsymbol{\mu}^a, \boldsymbol{\mu}^s, \boldsymbol{\sigma}, \boldsymbol{v}, \boldsymbol{\beta}$ excluding the kth or (j,k)th element at time t. These collections are used to obtain five conditional distributions:

- $p(\beta_t^{jk}|\mathcal{D}, \boldsymbol{\mu}^a, \boldsymbol{\mu}^s, \boldsymbol{\sigma}^2, \boldsymbol{v}^2, \boldsymbol{\beta}_{-t,jk}) = \mathcal{N}(\tilde{\beta}_t^{jk}, (\tilde{\delta}_{t,jk}^{(\beta)})^2)$, where

$$\begin{aligned}
\tilde{\beta}_t^{jk} &= \frac{(\delta_{t,jk}^{(\beta)})^2 \sum_{i=1}^{R} \alpha_t^{j(i)} m_{t+1,jk}^{(i)} + (v_{t+1}^k)^2 \beta_t^{ij(0)}}{(\delta_{t,jk}^{(\beta)})^2 \sum_{i=1}^{R} (\alpha_t^{j(i)})^2 + (v_{t+1}^k)^2} \\
\tilde{\delta}_{t,jk}^{(\beta)2} &= \frac{(\delta_{t,jk}^{(\beta)})^2 (v_{t+1}^k)^2}{(\delta_{t,jk}^{(\beta)})^2 \sum_{i=1}^{R} (\alpha_t^{j(i)})^2 + (v_{t+1}^k)^2}
\end{aligned}$$

  with $\alpha_t^{j(i)} = x_t^{j(i)} - \mu_t^j$ and $m_{t+1,jk}^{(i)} = (s_{t+1}^{k(i)} - \mu_{t+1}^k) - \sum_{X_t^\ell \in Pa(s_{t+1}^k)/\{x_t^j\}} \beta_t^{\ell k}(x_t^{\ell(i)} - \mu_t^\ell)$.

- $p\left((v_t^k)^2|\mathcal{D}, \boldsymbol{\mu}^a, \boldsymbol{\mu}^s, \boldsymbol{\sigma}^2, \boldsymbol{v}^2_{-t,k}, \boldsymbol{\beta}\right) = \text{Inv-}\Gamma\left(\frac{\tilde{\kappa}_{t,k}^{(v)}}{2}, \frac{\tilde{\rho}_{t,k}^{(v)}}{2}\right)$, where

$$\begin{aligned}
\tilde{\kappa}_{t,k}^{(v)} &= \kappa_{t,k}^{(v)} + R, \qquad \tilde{\rho}_{t,k}^{(v)} = \rho_{t,k}^{(v)} + \sum_{i=1}^{R} u_{t,k}^{(i)2}, \\
u_{t,k}^{(i)} &= (s_t^{k(i)} - \mu_t^k) - \sum_{X_t^j \in Pa(s_t^k)} \beta_t^{jk}(x_t^{j(i)} - \mu_t^j).
\end{aligned}$$

- $p\left((\sigma_t^k)^2|\mathcal{D},\boldsymbol{\mu}^a,\boldsymbol{\mu}^s,\boldsymbol{\sigma}^2_{-t,k},\boldsymbol{v}^2,\boldsymbol{\beta}\right) = \text{Inv-}\Gamma\left(\frac{\tilde{\kappa}^{(\sigma)}_{t,k}}{2},\frac{\tilde{\rho}^{(\sigma)}_{t,k}}{2}\right)$, where

$$\tilde{\kappa}^{(\sigma)}_{t,k} = \kappa^{(\sigma)}_{t,k} + R, \quad \tilde{\rho}^{(\sigma)}_{t,k} = \rho^{(\sigma)}_{t,k} + \sum_{i=1}^{R} u^{(i)2}_{t,k}, \quad u^{(i)}_{t,k} = s^{k(i)}_t - \mu^k_t.$$

- $p\left(\mu^k_t|\mathcal{D},\boldsymbol{\mu}^a,\boldsymbol{\mu}^s_{-t,k},\boldsymbol{\sigma}^2,\boldsymbol{v}^2,\boldsymbol{\beta}\right) = \mathcal{N}(\tilde{\mu}^k_t,(\tilde{\delta}^{(\mu)}_{t,k})^2)$, where

$$\begin{aligned}
\tilde{\mu}^k_t &= (\tilde{\delta}^{(\mu)}_{t,k})^2\left[\frac{\mu^{k(0)}_t}{(\delta^{(\mu)}_{t,k})^2} + \sum_{i=1}^{R}\frac{a^{(i)}_{t,k}}{(v^k_t)^2} + \sum_{i=1}^{R}\sum_{s^\ell_{t+1}\in Ch(s^k_t)}\frac{\beta^{k\ell}_t c^{(i)}_{t,k\ell}}{(v^\ell_t)^2}\right] \\
\frac{1}{(\tilde{\delta}^{(\mu)}_{t,k})^2} &= \frac{1}{(\delta^{(\mu)}_{t,k})^2} + \frac{R}{(v^k_t)^2} + \sum_{s^\ell_{t+1}\in Ch(s^k_t)}\frac{R(\beta^{k\ell}_t)^2}{(v^\ell_t)^2},
\end{aligned}$$

with

$$\begin{aligned}
a^{(i)}_{t,k} &= s^{k(i)}_t - \sum_{X^j_{t-1}\in Pa(s^k_t)}\beta^{jk}_{t-1}(x^{j(i)}_{t-1} - \mu^j_{t-1}) \\
c^{(i)}_{t,k\ell} &= \beta^{k\ell}_t s^{k(i)}_t - (s^{\ell(i)}_{t+1} - \mu^\ell_{t+1}) + \sum_{X^j_t\in Pa(s^\ell_{t+1})/\{s^k_t\}}\beta^{j\ell}_t(x^{j(i)}_t - \mu^j_t).
\end{aligned}$$

- $p\left(\lambda^k_t|\mathcal{D},\boldsymbol{\mu}^a_{-t,k},\boldsymbol{\mu}^s,\boldsymbol{\sigma}^2,\boldsymbol{v}^2,\boldsymbol{\beta}\right) = \mathcal{N}(\tilde{\lambda}^k_t,(\tilde{\delta}^{(\lambda)}_{t,k})^2)$, where

$$\begin{aligned}
\tilde{\lambda}^k_t &= (\tilde{\delta}^{(\lambda)}_{t,k})^2\left[\frac{\lambda^{k(0)}_t}{(\delta^{(\lambda)}_{t,k})^2} + \sum_{i=1}^{R}\frac{a^{k(i)}_t}{(v^k_t)^2} + \sum_{i=1}^{R}\sum_{s^\ell_{t+1}\in Ch(a^k_t)}\frac{\beta^{k\ell}_t c^{(i)}_{t,k\ell}}{(v^\ell_t)^2}\right] \\
\frac{1}{(\tilde{\delta}^{(\lambda)}_{t,k})^2} &= \frac{1}{(\delta^{(\lambda)}_{t,k})^2} + \frac{R}{(v^k_t)^2} + \sum_{s^\ell_{t+1}\in Ch(a^k_t)}\frac{R(\beta^{k\ell}_t)^2}{(v^\ell_t)^2}
\end{aligned}$$

with

$$c^{(i)}_{t,k\ell} = \beta^{k\ell}_t a^{k(i)}_t - (s^{\ell(i)}_{t+1} - \mu^\ell_{t+1}) + \sum_{X^j_t\in Pa(s^\ell_{t+1})/\{a^k_t\}}\beta^{j\ell}_t(x^{j(i)}_t - \mu^j_t).$$

In Gibbs sampling, we set a prior $p(\boldsymbol{w})$ and sample $\boldsymbol{w}^{(0)}$ from it. Now, given $\boldsymbol{w}^{(i-1)}$ we sequentially compute and generate one sample from the above conditional posterior distributions for each parameter, obtaining a new $\boldsymbol{w}^{(i)}$. By repeating this process, one can arrive at a sample whose distribution is a very close approximation to the desired posterior.

9 Kinetic Modeling of Fed-Batch Fermentation of Yarrowia lipolytica

Cell growth and citrate production of *Yarrowia lipolytica* take place inside a bioreactor, which requires carbon sources (e.g., soybean oil or waste cooking oil), nitrogen (yeast extract and ammonia sulfate), and oxygen, subject to good mixing and carefully controlled operating conditions (e.g., pH, temperature). During fermentation, the feed rate is adjusted to control the concentration of oil.

In our case study, we treat feed rate as the only control. Other CPPs relevant to the bioprocess are set automatically as follows. The dissolved oxygen level is set at 30% of air saturation using cascade controls of agitation speed between 500 and 1,400 rpm, with the aeration rate fixed at 0.3 L/min. The pH is controlled at 6.0 for the first 12 hours, then maintained at 7.0 until the end of the process. The temperature is maintained at $30°$C for the entire run. We stop the fermentation process after 140 hours (or if the volume reaches the bioreactor capacity).

The kinetics of the fed-batch fermentation of *Yarrowia lipolytica* can be described by a system of differential equations. The biomanufacturing literature uses deterministic ODEs, but we have augmented some of the equations with Brownian noise terms of the form dB_t to reflect the inherent stochastic uncertainty of the bioprocess. Table 4 provides a list of environmental parameters that appear in the equations, together with their values in our case study. The equations themselves can be grouped into eight parts as follows:

Table 4: Mechanistic model parameter description and estimation.

Parameters	Descrption	Estimation	Unit
α_L	Coefficient of lipid production for cell growth	0.1273	-
C_{max}	Maximum citrate concentration that cells can tolerate	130.90	g/L
K_{iN}	Nitrogen limitation constant to trigger on lipid and citrate production	0.1229	g/L
K_{iS}	Inhibition constant for substrate in lipid-based growth kinetics	612.18	g/L
K_{iX}	Constant for cell density effect on cell growth and lipid/citrate formation	59.974	g/L
K_N	Saturation constant for intracellular nitrogen in growth kinetics	0.0200	g/L
K_O	Saturation constant for dissolved oxygen in kinetics of cell growth, substrate uptake, lipid consumption by β-oxidation	0.3309	% Air
K_S	Saturation constant for substrate utilization	0.0430	g/L
K_{SL}	Coefficient for lipid consumption/decomposition	0.0217	-
m_s	Maintenance coefficient for substrate	0.0225	g/g/h
r_L	Constant ratio of lipid carbon flow to total carbon flow (lipid + citrate)	0.4792	-
V_{evap}	Evaporation rate (or loss of volume) in the fermentation	0.0026	L/h
Y_{cs}	Yield coefficient of citrate based on substrate consumed	0.6826	g/g
Y_{ls}	Yield coefficient of lipid based on substrate consumed	0.3574	g/g
Y_{xn}	Yield coefficient of cell mass based on nitrogen consumed	10.0	g/g
Y_{xs}	Yield coefficient of cell mass based on substrate consumed	0.2386	g/g
β_{LCmax}	Coefficient of maximum carbon flow for citrate and lipid	0.1426	h-1
μ_{max}	Maximum specific growth rate on substrate	0.3845	h-1
S_F	Oil concentration in oil feed	917.00	g/L

1. **Cell mass**: The total cell mass X consists of lipid-free (X_f) and lipid (L) mass, and is measured by dry cell weight (DCW) in fermentation experiments.

$$X = X_f + L$$

2. **Dilution rate**: The dilution D of the working liquid volume in the bioreactor is caused by feed of base, such as KOH solution (F_B) and substrate (F_S):

$$D = \frac{F_B + F_S}{V}$$

3. **Lipid-free cell growth**: Cell growth consumes nutrients, including the substrate (carbon source) S, nitrogen N, and dissolved oxygen O, and is described by coupled Monod equations, with considerations of inhibitions from high oil concentrations and cell densities:

$$\mathrm{d}X_f = \mu X_f \mathrm{d}t - \left(D - \frac{V_{evap}}{V}\right) X_f \mathrm{d}t + \sigma(X_f)\mathrm{d}B_t$$
$$\mu = \mu_{max}\left(\frac{S}{K_S + S}\cdot\frac{1}{1 + S/K_{iS}}\right)\frac{N}{K_N + N}\cdot\frac{O}{K_O + O}\cdot\frac{1}{1 + X_f^{t_1}/K_{ix}}$$

4. **Citrate accumulation**: Citrate (C) is an overflow of all the carbon introduced to the lipid synthesis pathway (β_{LC}), which is more active under nitrogen-limited, substrate-rich, and aerobic conditions. Only a proportion r_L of the total carbon flow (citrate plus lipid) in the lipid synthesis pathway goes to production due to the overflow loss in citrate. A tolerance limit C_{max} of citrate, and the effect of cell density on product formation ($1/(1 + X_f/K_{iX})$), are also considered in the model.

$$\mathrm{d}C = \beta_C \cdot X_f \mathrm{d}t - \left(D - \frac{V_{evap}}{V}\right) C\mathrm{d}t + \sigma(C)\mathrm{d}B_t$$
$$\beta_C = 2(1 - r_L)\beta_{LC}$$
$$\beta_{LC} = \frac{1}{1 + N/K_{iN}}\cdot\left(\frac{S}{K_S + S}\cdot\frac{1}{1 + S/K_{iS}}\right)\frac{O}{K_O + O}\cdot\frac{1}{1 + X_f/K_{iX}}\left(1 - \frac{C}{C_{max}}\right)\beta_{LCmax}$$

5. **Lipid accumulation**: Lipid is accumulated under nitrogen-limited, substrate-rich, and aerobic conditions. Lipid production is described using partial growth-association kinetics; a small portion of lipid can be degraded

when its concentration is high in the presence of oxygen.

$$\begin{aligned}\mathrm{d}L &= q_L \cdot X_f \mathrm{d}t - \left(D - \frac{V_{evap}}{V}\right) L\mathrm{d}t + \sigma(L)\mathrm{d}B_t \\ &= \left(\alpha_L \cdot \mu + \beta_L\right) X_f \mathrm{d}t - \left(D - \frac{V_{evap}}{V}\right) L\mathrm{d}t + \sigma(L)\mathrm{d}B_t \\ \beta_L &= r_L \cdot \beta_{LC} - K_{SL}\frac{L}{L+X_f} \cdot \frac{O}{K_O + O}\end{aligned}$$

6. **Substrate consumption**: Oil (S) is fed during the fed-batch fermentation and used for cell growth, energy maintenance, citrate formation, and lipid production.

$$\begin{aligned}-dS &= q_S \cdot X_f \mathrm{d}t - \frac{F_S}{V} S_F \mathrm{d}t + \left(D - \frac{V_{evap}}{V}\right) S\mathrm{d}t + \sigma(S)\mathrm{d}B_t \\ q_S &= \frac{1}{Y_{X/S}}\mu + \frac{O}{K_O + O} \cdot \frac{S}{K_S + S} m_S + \frac{1}{Y_{C/S}}\beta_C + \frac{1}{Y_{L/S}}\beta_L\end{aligned}$$

7. **Nitrogen consumption**: Extracellular nitrogen (N) is used for cell growth.

$$-dN = \frac{1}{Y_{X/N}}\mu X_f \mathrm{d}t + (D - \frac{V_{evap}}{V})N\mathrm{d}t + \sigma(N)\mathrm{d}B_t$$

8. **Volume change**: The rate V of working volume change of the fermentation is calculated based on the rates of base feed (F_B), substrate feed (F_S), and evaporation (V_{evap}).

$$\begin{aligned}dV &= (F_B + F_S - V_{evap})\mathrm{d}t + \sigma(V)\mathrm{d}B_t \\ F_B &= \frac{V}{1000}\left(\frac{7.14}{Y_{X/N}}\mu X_f + 1.59\beta_C X_f\right)\end{aligned}$$

Figure 4 in the main text shows the fit of the ODE trajectory to the 8 available batches of experimental data. The fit is not perfect due to the limited sample size and the noticeably large variation between experimental trajectories (a consequence of the fact that scientists conducted these experiments with very different oil feeding profiles).

10 Proofs

Below, we give full proofs for all results that were stated in the main text.

10.1 Proof for Proposition 1

By plugging in the linear policy function (10) into the state transition in (5), we have

$$\begin{aligned}\boldsymbol{s}_{t+1} &= \boldsymbol{\mu}^s_{t+1} + (\boldsymbol{\beta}^s_t)^\top(\boldsymbol{s}_t - \boldsymbol{\mu}^s_t) + (\boldsymbol{\beta}^a_t)^\top(\boldsymbol{\vartheta}^\top_t(\boldsymbol{s}_t) - \boldsymbol{\mu}^a_t) + V_{t+1}\mathbf{z} \\ &= \boldsymbol{\mu}^s_{t+1} + (\boldsymbol{\beta}^s_t)^\top(\boldsymbol{s}_t - \boldsymbol{\mu}^s_t) + (\boldsymbol{\beta}^a_t)^\top\left[\boldsymbol{\mu}^a_t + \boldsymbol{\vartheta}^\top_t(\boldsymbol{s}_t - \boldsymbol{\mu}^t_s) - \boldsymbol{\mu}^a_t\right] + \boldsymbol{e}_{t+1} \\ &= \boldsymbol{\mu}^s_{t+1} + \left((\boldsymbol{\beta}^s_t)^\top + (\boldsymbol{\beta}^a_t)^\top\boldsymbol{\vartheta}^\top_t\right)(\boldsymbol{s}_t - \boldsymbol{\mu}^s_t) + \boldsymbol{e}_{t+1} \qquad (21) \\ &= \boldsymbol{\mu}^s_{t+1} + ((\boldsymbol{\beta}^s_t)^\top + (\boldsymbol{\beta}^a_t)^\top\boldsymbol{\vartheta}^\top_t)\left(\left(\boldsymbol{\beta}^s_{t-1}\right)^\top + \left(\boldsymbol{\beta}^a_{t-1}\right)^\top\boldsymbol{\vartheta}^\top_{t-1}\right)(\boldsymbol{s}_{t-1} - \boldsymbol{\mu}^s_{t-1}) + \left[((\boldsymbol{\beta}^s_t)^\top + (\boldsymbol{\beta}^a_t)^\top\boldsymbol{\vartheta}^\top_t)\boldsymbol{e}_t + \boldsymbol{e}_{t+1}\right] \\ &= \boldsymbol{\mu}^s_{t+1} + \mathbf{R}_{t-1,t}(\boldsymbol{s}_{t-1} - \boldsymbol{\mu}^s_{t-1}) + \left[\left((\boldsymbol{\beta}^s_t)^\top + (\boldsymbol{\beta}^a_t)^\top\boldsymbol{\vartheta}^\top_t\right)\boldsymbol{e}_t + \boldsymbol{e}_{t+1}\right] \\ &\quad \cdots \\ &= \boldsymbol{\mu}^s_{t+1} + \prod_{i=1}^{t}\left((\boldsymbol{\beta}^s_i)^\top + (\boldsymbol{\beta}^a_i)^\top\boldsymbol{\vartheta}^\top_i\right)(\boldsymbol{s}_1 - \boldsymbol{\mu}^s_1) + \left[\sum_{i=1}^{t}\left(\prod_{j=i}^{t}\left(\left(\boldsymbol{\beta}^s_j\right)^\top + \left(\boldsymbol{\beta}^a_j\right)^\top\boldsymbol{\vartheta}^\top_j\right)\boldsymbol{e}_i\right) + \boldsymbol{e}_{t+1}\right] \qquad (22) \\ &= \boldsymbol{\mu}^s_{t+1} + \mathbf{R}_{1,t}(\boldsymbol{s}_1 - \boldsymbol{\mu}^s_1) + \left[\sum_{i=1}^{t}\mathbf{R}_{i,t}\boldsymbol{e}_i + \boldsymbol{e}_{t+1}\right] \qquad (23)\end{aligned}$$

where $\boldsymbol{e}_t$ is a n-dimensional normal random column vector with mean zero and diagonal covariance matrix $V_t \triangleq \mathrm{diag}((v^s_t)^2)$. Similarly, one can expand $\boldsymbol{s}_{t+1}$ until time h to obtain an analog of (22), which yields the desired result.

10.2 Proof for Proposition 2

The conditional mean follows directly from (23) above. Because the vectors $\boldsymbol{e}_i, \boldsymbol{e}_j$ are mutually independent, we obtain

$$\begin{aligned}
\text{Var}[\boldsymbol{s}_{t+1}|\boldsymbol{s}_1, \boldsymbol{\pi}_{\boldsymbol{\theta}}] &= \text{Var}\left[\sum_{i=1}^{t} \mathbf{R}_{i,t}\boldsymbol{e}_i + \boldsymbol{e}_{t+1}\right] \\
&= \sum_{i=1}^{t} \mathbf{R}_{i,t}\text{Var}\left[\boldsymbol{e}_i\right]\mathbf{R}_{i,t}^{\top} + \text{Var}\left[\boldsymbol{e}_{t+1}\right] \\
&= \sum_{i=1}^{t} \mathbf{R}_{i,t}V_i\mathbf{R}_{i,t}^{\top} + V_{t+1}.
\end{aligned}$$

The objective function can be obtained from (6) and (10) as,

$$\begin{aligned}
J(\boldsymbol{\theta}; \boldsymbol{w}) &= \sum_{t=1}^{H} \mathbb{E}[r_t(\boldsymbol{s}_t, \boldsymbol{a}_t)|\boldsymbol{\pi}_{\boldsymbol{\theta}}, \boldsymbol{s}_1, \boldsymbol{w}] \\
&= \sum_{t=1}^{H} m_t + \boldsymbol{b}_t^{\top}\boldsymbol{a}_t + \boldsymbol{c}_t^{\top}\mathbb{E}[\boldsymbol{s}_t|\boldsymbol{\pi}_{\boldsymbol{\theta}}, \boldsymbol{s}_1, \boldsymbol{w}] \\
&= \sum_{t=1}^{H} m_t + \boldsymbol{b}_t^{\top}\boldsymbol{\mu}_t^a + (\boldsymbol{b}_t^{\top}\boldsymbol{\vartheta}_t^{\top} + \boldsymbol{c}_t^{\top})\mathbb{E}[\boldsymbol{s}_t|\boldsymbol{s}_1, \boldsymbol{\pi}_{\boldsymbol{\theta}}] - \boldsymbol{b}_t^{\top}\boldsymbol{\vartheta}^{\top}\boldsymbol{\mu}_t^s.
\end{aligned}$$

Plugging in the conditional mean computed previously, we obtain

$$\begin{aligned}
J(\boldsymbol{\theta}; \boldsymbol{w}) &= \sum_{t=1}^{H} m_t + \boldsymbol{b}_t^{\top}\boldsymbol{\mu}_t^a + (\boldsymbol{b}_t^{\top}\boldsymbol{\vartheta}_t^{\top} + \boldsymbol{c}_t^{\top})(\boldsymbol{\mu}_t^s + \mathbf{R}_{1,t}(\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s)) - \boldsymbol{b}_t^{\top}\boldsymbol{\vartheta}_t^{\top}\boldsymbol{\mu}_t^s \\
&= \sum_{t=1}^{H} m_t + \boldsymbol{b}_t^{\top}\boldsymbol{\mu}_t^a + \boldsymbol{c}_t^{\top}\boldsymbol{\mu}_t^s + (\boldsymbol{b}_t^{\top}\boldsymbol{\vartheta}_t^{\top} + \boldsymbol{c}_t^{\top})\mathbf{R}_{1,t}(\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s).
\end{aligned}$$

10.3 Proof of Theorem 1

From Proposition 1, we have $\bar{\boldsymbol{s}}_t = \boldsymbol{\mu}_t^s + \mathbf{R}_{h,t-1}(\bar{\boldsymbol{s}}_h - \boldsymbol{\mu}_h^s)$. We then take the gradient of $\bar{\boldsymbol{s}}_t$ over the policy parameters $\boldsymbol{\vartheta}_h$, obtaining

$$\begin{aligned}
\frac{\partial \bar{\boldsymbol{s}}_t}{\partial \boldsymbol{\vartheta}_h} &= \frac{\partial}{\partial \boldsymbol{\vartheta}_h}\mathbf{R}_{h,t-1}(\bar{\boldsymbol{s}}_h - \boldsymbol{\mu}_h^s) \\
&= (\bar{\boldsymbol{s}}_h - \boldsymbol{\mu}_h^s)^{\top}\left[\prod_{i=h+1}^{t-1}(\boldsymbol{\beta}_i^s + \boldsymbol{\vartheta}_i\boldsymbol{\beta}_i^a)^{\top}\right]\boldsymbol{\beta}_h^{a\top} \\
&= (\bar{\boldsymbol{s}}_h - \boldsymbol{\mu}_h^s)^{\top}\mathbf{R}_{h+1,t-1}\left(\boldsymbol{\beta}_h^a\right)^{\top}. \qquad (24)
\end{aligned}$$

Combining this derivation with (14), we can obtain $\frac{\partial \bar{r}_t}{\partial \boldsymbol{\vartheta}_t}$ in the following cases:

- When $h = t$,

$$\frac{\partial \bar{r}_t}{\partial \boldsymbol{\vartheta}_t} = \frac{\partial}{\partial \boldsymbol{\vartheta}_t}\left(m_t + \boldsymbol{c}_t^{\top}\bar{\boldsymbol{s}}_t + \boldsymbol{b}_t^{\top}(\boldsymbol{\mu}_t^a + \boldsymbol{\vartheta}_t^{\top}(\bar{\boldsymbol{s}}_t - \boldsymbol{\mu}_t^s))\right) = (\bar{\boldsymbol{s}}_t - \boldsymbol{\mu}_t^s)\,\boldsymbol{b}_t^{\top} \qquad (25)$$

- When $h = t - 1$,

$$\begin{aligned}
\frac{\partial \bar{r}_t}{\partial \boldsymbol{\vartheta}_{t-1}} &= \frac{\partial}{\partial \boldsymbol{\vartheta}_{t-1}}\left(m_t + \boldsymbol{c}_t^{\top}\bar{\boldsymbol{s}}_t + \boldsymbol{b}_t^{\top}(\boldsymbol{\mu}_t^a + \boldsymbol{\vartheta}_t^{\top}(\bar{\boldsymbol{s}}_t - \boldsymbol{\mu}_t^s))\right) \\
&= \frac{\partial}{\partial \boldsymbol{\vartheta}_{t-1}}(\boldsymbol{c}_t^{\top}\bar{\boldsymbol{s}}_t + \boldsymbol{b}_t^{\top}\boldsymbol{\vartheta}_t^{\top}\bar{\boldsymbol{s}}_t) \\
&= (\boldsymbol{c}_t + \boldsymbol{\vartheta}_t\boldsymbol{b}_t)\frac{\partial \bar{\boldsymbol{s}}_t}{\partial \boldsymbol{\vartheta}_{t-1}} \\
&= (\boldsymbol{c}_t + \boldsymbol{\vartheta}_t\boldsymbol{b}_t)(\bar{\boldsymbol{s}}_{t-1} - \boldsymbol{\mu}_{t-1}^s)^{\top}\left(\boldsymbol{\beta}_{t-1}^a\right)^{\top}, \qquad (26)
\end{aligned}$$

  where (26) follows by applying (21) to $\boldsymbol{s}_t$.

- When $h < t-1$,

$$\begin{aligned}\frac{\partial \bar{r}_t}{\partial \boldsymbol{\vartheta}_k} &= \frac{\partial}{\partial \boldsymbol{\vartheta}_t}\left(m_t + \boldsymbol{c}_t^\top \bar{\boldsymbol{s}}_t + \boldsymbol{b}_t^\top(\boldsymbol{\mu}_t^a + \boldsymbol{\vartheta}_t^\top(\bar{\boldsymbol{s}}_t - \boldsymbol{\mu}_t^s))\right)\\ &= (\boldsymbol{c}_t + \boldsymbol{\vartheta}_t \boldsymbol{b}_t)\frac{\partial \bar{\boldsymbol{s}}_t}{\partial \boldsymbol{\vartheta}_h}\\ &= (\boldsymbol{c}_t + \boldsymbol{\vartheta}_t \boldsymbol{b}_t)(\bar{\boldsymbol{s}}_h - \boldsymbol{\mu}_h^s)^\top \left[\prod_{i=h+1}^{t-1}(\boldsymbol{\beta}_i^s + \boldsymbol{\vartheta}_i \boldsymbol{\beta}_i^a)^\top\right](\boldsymbol{\beta}_h^a)^\top, \qquad (27)\end{aligned}$$

  where (27) follows by applying (24).

The desired conclusion follows from (25)-(27).

10.4 Proof of Proposition 3

Recall that the multiplication of two matrices $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{m\times p}$ requires $\mathcal{O}(nmp)$ time. In our setting, we are given the model parameters $\boldsymbol{\mu}_t^s \in \mathbb{R}^n$, $\boldsymbol{\mu}_t^a \in \mathbb{R}^m$, $\boldsymbol{\beta}_t^s \in \mathbb{R}^{n\times n}$, $\boldsymbol{\beta}_t^a \in \mathbb{R}^{m\times n}$, as well as the policy parameters $\vartheta_t \in \mathbb{R}^{n\times m}$, $\boldsymbol{b}_t \in \mathbb{R}^m$ and $\boldsymbol{c}_t \in \mathbb{R}^n$. We also have the expected state $\bar{\boldsymbol{s}}_t \in \mathbb{R}^n$.

We now consider the computation of $\frac{\partial \bar{r}_t}{\partial \boldsymbol{\vartheta}_h}$. If $t = h$, both nested backpropagation (NBP) and brute force require $\mathcal{O}(nm)$ time to compute $(\bar{\boldsymbol{s}}_t - \boldsymbol{\mu}_t^s)\boldsymbol{b}_t^\top$. For each t and $h < t$, it takes n^2 addition operations and $n^2 m$ multiplication operations to compute $\boldsymbol{\beta}_i^s + \boldsymbol{\vartheta}_i \boldsymbol{\beta}_i^a$. Therefore, the brute-force approach, which directly computes $\boldsymbol{\delta}_h^t$ in (15), takes

$$\mathcal{O}\left((t-h)(n^2 + n^2 m) + n^2 m\right) = \mathcal{O}\left((t-h)(n^2 + n^2 m)\right)$$

time when $t \neq h$. Then, to compute the gradient with respect to each parameter in the network, it takes

$$\begin{aligned}\mathcal{O}\left(\sum_{t=1}^{H}\sum_{h=1}^{t-1}(t-h)(n^2+n^2m)\right) &= \mathcal{O}\left(\sum_{t=1}^{H}\frac{t(t-1)}{2}(n^2+n^2m)\right)\\ &= \mathcal{O}\left(\frac{H^3-H}{2}(n^2+n^2m)\right)\\ &= \mathcal{O}\left(H^3(n^2m)\right).\end{aligned}$$

In comparison, by storing and reusing the computation of $\mathbf{R}_{h,t-1}$ and $\boldsymbol{\beta}_i^s + \boldsymbol{\vartheta}_i \boldsymbol{\beta}_i^a$ (see Step 1 in Algorithm 1), NBP takes $\mathcal{O}(n^2 m)$ time to compute $\boldsymbol{\delta}_h^t$ and $\mathcal{O}(n^2 + n^2 m)$ time to compute $\mathbf{R}_{h,t}$. Thus, the total computational cost of NBP becomes

$$\mathcal{O}\left(\sum_{t=1}^{H}\sum_{h=1}^{h}(n^2+n^2m) + \sum_{t=1}^{H}\sum_{h=1}^{h}n^2 m\right) = \mathcal{O}\left(H^2 n^2 m\right).$$

10.5 Proof of Lemma 1

In the following, we will use $\|A\|_F$ to denote the Frobenius norm of a matrix A. A useful property of this norm is $\|AB\|_F \leq \|A\|_F \cdot \|B\|_F$ for matrices A, B. We also use the notation $|A|$ to represent a matrix whose elements are equal to the absolute values of the corresponding elements of A.

Fix $\boldsymbol{w} \in \mathcal{W}$. From (11), the gradient of the objective function can be expressed as

$$\frac{\partial J(\boldsymbol{\theta};\boldsymbol{w})}{\partial \boldsymbol{\vartheta}_t} = \mathbf{R}_{1,t}(\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s)\boldsymbol{b}_t^\top + \sum_{t'=t}^{H}(\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s)(\boldsymbol{b}_t^\top \boldsymbol{\vartheta}_t^\top + \boldsymbol{c}_t^\top)\frac{\partial \mathbf{R}_{1,t'}}{\partial \boldsymbol{\vartheta}_t}, \qquad (28)$$

where $\frac{\partial \mathbf{R}_{1,t'}}{\partial \boldsymbol{\vartheta}_t} = \left(\prod_{i=1;i\neq t}^{t'}\left(\boldsymbol{\beta}_i^{s\top} + \boldsymbol{\beta}_i^{a\top}\boldsymbol{\vartheta}_i^\top\right)\right)\boldsymbol{\beta}_t^{a\top}$.

The proof proceeds in three steps: we show that $\mathcal{J}$ is differentiable; that $\frac{\partial J(\boldsymbol{\theta};\boldsymbol{w})}{\partial \vartheta_t}$ is L-smooth; and that $\mathcal{J}$ itself is L-smooth.

Step 1: $\mathcal{J}$ is differentiable.

Equation (28) shows that $J(\boldsymbol{\theta};\boldsymbol{w})$ is differentiable for any $\boldsymbol{w}$ (note that, if $\boldsymbol{w} \notin \mathcal{W}$, the derivative is zero). Since $\mathbb{C}$ is bounded, we can let $B_{\mathbb{C}} \equiv \max_{\boldsymbol{x}\in\mathbb{C}}\|\boldsymbol{x}\|_F$. Let $B \in \mathbb{R}^{n\times m}$ be a matrix with all entries equal to $B_{\mathbb{C}}$.

The function

$$
\begin{aligned}
g(\boldsymbol{w}) &= \prod_{i=1}^{t}\left(|\boldsymbol{\beta}_i^s|^\top + |\boldsymbol{\beta}_i^a|^\top B^\top\right)|\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s|\cdot|\boldsymbol{b}_t|^\top \\
&\quad + \sum_{t'=t}^{H}|\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s|\left(|\boldsymbol{b}_t|^\top B^\top + |\boldsymbol{c}_t|^\top\right)\left(\prod_{i=1;i\neq t}^{t'}\left(|\boldsymbol{\beta}_i^s|^\top + |\boldsymbol{\beta}_i^a|^\top B^\top\right)\right)|\boldsymbol{\beta}_t^a|^\top
\end{aligned}
$$

is non-negative and Lebesgue integrable on $\mathcal{W}$. We have $|\frac{\partial}{\partial\boldsymbol{\theta}}J(\boldsymbol{\theta};\boldsymbol{w})| \leq g(\boldsymbol{w})$ for all $(\boldsymbol{\theta},\boldsymbol{w}) \in \mathbb{C}\times\mathcal{W}$. Then,

$$
\begin{aligned}
\nabla\mathcal{J}(\boldsymbol{\theta}) &= \frac{\partial}{\partial\boldsymbol{\theta}}\int J(\boldsymbol{\theta};\boldsymbol{w})p(\boldsymbol{w}|\mathcal{D})d\boldsymbol{w} \\
&= \int\frac{\partial}{\partial\boldsymbol{\theta}}J(\boldsymbol{\theta};\boldsymbol{w})p(\boldsymbol{w}|\mathcal{D})d\boldsymbol{w} \qquad (29)\\
&= \int_{\mathcal{W}}\frac{\partial}{\partial\boldsymbol{\theta}}J(\boldsymbol{\theta};\boldsymbol{w})p(\boldsymbol{w}|\mathcal{D})d\boldsymbol{w}
\end{aligned}
$$

where (29) follows by the dominated convergence theorem. We conclude that $\mathcal{J}$ is differentiable.

Step 2: The gradient of $J(\boldsymbol{\theta};w)$ is L-smooth in $\boldsymbol{\theta}$.

Fix $\boldsymbol{x} = (\boldsymbol{x}_1,\ldots,\boldsymbol{x}_{H-1}) \in \mathbb{C}$ and $\boldsymbol{y} = (\boldsymbol{y}_1,\ldots,\boldsymbol{y}_{H-1}) \in \mathbb{C}$ such that $\boldsymbol{x}_i = \boldsymbol{y}_i$ for $i \neq t$ and $\boldsymbol{x}_t \neq \boldsymbol{y}_t$. Without loss of generality, we may assume $\boldsymbol{w} \in \mathcal{W}$. We derive

$$
\begin{aligned}
&\left\|\frac{\partial J(\boldsymbol{x};\boldsymbol{w})}{\partial\boldsymbol{\vartheta}_t} - \frac{\partial J(\boldsymbol{y};\boldsymbol{w})}{\partial\boldsymbol{\vartheta}_t}\right\|_F \\
=\ &\left\|\left(\mathbf{R}_{1,t}(\boldsymbol{x}) - \mathbf{R}_{1,t}(\boldsymbol{y})\right)(\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s))\boldsymbol{b}_t^\top + \sum_{t'=t}^{H}(\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s)(\boldsymbol{b}_t^\top\boldsymbol{x}_t^\top + \boldsymbol{c}_t^\top)\frac{\partial\mathbf{R}_{1,t'}(\boldsymbol{x})}{\partial\boldsymbol{\vartheta}_t}\right. \\
&\left. - \sum_{t'=t}^{H}(\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s)(\boldsymbol{b}_t^\top\boldsymbol{y}_t^\top + \boldsymbol{c}_t^\top)\frac{\partial\mathbf{R}_{1,t'}(\boldsymbol{y})}{\partial\boldsymbol{\vartheta}_t}\right\|_F \\
=\ &\left\|\left(\mathbf{R}_{1,t}(\boldsymbol{x}) - \mathbf{R}_{1,t}(\boldsymbol{y})\right)(\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s)\boldsymbol{b}_t^\top + (\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s)(\boldsymbol{b}_t^\top\boldsymbol{x}_t^\top + \boldsymbol{c}_t^\top)\sum_{t'=t}^{H}\frac{\partial\mathbf{R}_{1,t'}(\boldsymbol{x})}{\partial\boldsymbol{\vartheta}_t}\right. \\
&\left. -(\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s)(\boldsymbol{b}_t^\top\boldsymbol{y}_t^\top + \boldsymbol{c}_t^\top)\sum_{t'=t}^{H}\frac{\partial\mathbf{R}_{1,t'}(\boldsymbol{x})}{\partial\boldsymbol{\vartheta}_t}\right\|_F \\
\leq\ &\left\|\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s\boldsymbol{b}_t^\top\right\|_F\left\|\mathbf{R}_{1,t}(\boldsymbol{x}) - \mathbf{R}_{1,t}(\boldsymbol{y})\right\|_F + \left\|(\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s)\right\|_F\left\|\boldsymbol{b}_t^\top(\boldsymbol{x}_t^\top - \boldsymbol{y}_t^\top)\sum_{t'=t}^{H}\frac{\partial\mathbf{R}_{1,t'}(\boldsymbol{x})}{\partial\boldsymbol{\vartheta}_t}\right\|_F \qquad (30)\\
\leq\ &\|\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s\|_F\|\boldsymbol{b}_t\|_F\left\|\left(\prod_{i=1}^{t-1}\left((\boldsymbol{\beta}_i^s)^\top + (\boldsymbol{\beta}_i^a)^\top\boldsymbol{x}_i^\top\right)\right)(\boldsymbol{\beta}_t^a)^\top(\boldsymbol{x}_t - \boldsymbol{y}_t)^\top\right\|_F \\
&+ \|\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s\|_F\|\boldsymbol{b}_t\|_F\|\boldsymbol{x}_t - \boldsymbol{y}_t\|_F\left\|\sum_{t'=t}^{H}\frac{\partial\mathbf{R}_{1,t'}(\boldsymbol{x})}{\partial\boldsymbol{\vartheta}_t}\right\|_F. \\
\leq\ &\|\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s\|_F\|\boldsymbol{b}_t\|\left\|\prod_{i=1}^{t-1}\left((\boldsymbol{\beta}_i^s)^\top + (\boldsymbol{\beta}_i^a)^\top\boldsymbol{x}_i^\top\right)\right\|_F\|\boldsymbol{\beta}_t^a\|_F\left\|(\boldsymbol{x}_t - \boldsymbol{y}_t)^\top\right\|_F \\
&+ \|\boldsymbol{s}_1 - \boldsymbol{\mu}_1^s\|\|\boldsymbol{b}_t\|_F\|\boldsymbol{x}_t - \boldsymbol{y}_t\|_F\left\|\sum_{t'=t}^{H}\frac{\partial\mathbf{R}_{1,t'}(\boldsymbol{x})}{\partial\boldsymbol{\vartheta}_t}\right\|_F, \\
\leq\ &(\|\boldsymbol{s}_1\| + \|\boldsymbol{\mu}_1^s\|)\|\boldsymbol{b}_t\|_F\left\|\prod_{i=1}^{t-1}\left(\boldsymbol{\beta}_i^{s\top} + \boldsymbol{\beta}_i^{a\top}\boldsymbol{x}_i^\top\right)\right\|_F\|\boldsymbol{\beta}_t^a\|_F\left\|(\boldsymbol{x}_t - \boldsymbol{y}_t)^\top\right\|_F \\
&+(\|\boldsymbol{s}_1\| + \|\boldsymbol{\mu}_1^s\|)\|\boldsymbol{b}_t\|\|\boldsymbol{x}_t - \boldsymbol{y}_t\|_F\left\|\sum_{t'=t}^{H}\frac{\partial\mathbf{R}_{1,t'}(\boldsymbol{x})}{\partial\boldsymbol{\vartheta}_t}\right\|_F, \qquad (31)
\end{aligned}
$$

where (30) and (31) follow from the triangle inequality.

We can rewrite (31) as

$$\left\| \frac{\partial J(\boldsymbol{x};\boldsymbol{w})}{\partial \boldsymbol{\vartheta}_t} - \frac{\partial J(\boldsymbol{y};\boldsymbol{w})}{\partial \boldsymbol{\vartheta}_t} \right\|_F \leq L_t(\boldsymbol{w}) \left\| \boldsymbol{x}_t - \boldsymbol{y}_t \right\|_F \tag{32}$$

where

$$L_t(\boldsymbol{w}) = (\|\boldsymbol{s}_1\| + \|\boldsymbol{\mu}_1^s\|)\|\boldsymbol{b}_t\|_F \left\| \prod_{i=1}^{t-1} \left(\boldsymbol{\beta}_i^{s\top} + \boldsymbol{\beta}_i^{a\top} \boldsymbol{x}_i^\top \right) \right\|_F \|\boldsymbol{\beta}_t^a\|_F + (\|\boldsymbol{s}_1\| + \|\boldsymbol{\mu}_1^s\|) \|\boldsymbol{b}_t\|_F \left\| \sum_{t'=t}^{H} \frac{\partial \mathbf{R}_{1,t'}(\boldsymbol{x})}{\partial \boldsymbol{\vartheta}_t} \right\|_F$$

Since $\mathcal{W}$ is bounded, define $B_{\mathcal{W}} = \max_{\boldsymbol{w}\in\mathcal{W}} \|\boldsymbol{w}\|$ to be a constant bounding $\boldsymbol{w} \in \mathcal{W}$. Recall that we also have $\|\boldsymbol{x}_t\|_F \leq B_{\mathbb{C}}$. Then, by applying the triangle inequality, we have

$$\begin{aligned} \left\| \frac{\partial \mathbf{R}_{1,t'}(\boldsymbol{x})}{\partial \boldsymbol{\vartheta}_t} \right\|_F &\leq \left(\prod_{i=1;i\neq t}^{t'} (\|\boldsymbol{\beta}_i^s\|_F + \|\boldsymbol{\beta}_i^a\|_F \|\boldsymbol{\vartheta}_i\|_F) \right) \|\boldsymbol{\beta}_t^a\|_F \\ &\leq \left(\prod_{i=1;i\neq t}^{t'} (B_{\mathcal{W}} + B_{\mathcal{W}} B_{\mathbb{C}}) \right) B_{\mathcal{W}}, \end{aligned} \tag{33}$$

and

$$\left\| \prod_{i=1}^{t-1} \left(\boldsymbol{\beta}_i^{s\top} + \boldsymbol{\beta}_i^{a\top} \boldsymbol{x}_i^\top \right) \right\|_F \leq \prod_{i=1}^{t-1} (\|\boldsymbol{\beta}_i^s\|_F + \|\boldsymbol{\beta}_i^a\|_F \|\boldsymbol{\vartheta}_i\|_F) \leq \prod_{i=1}^{t-1} (B_{\mathcal{W}} + B_{\mathcal{W}} B_{\mathbb{C}}) . \tag{34}$$

By plugging (33) and (34) into (32), we bound

$$\begin{aligned} L_t(\boldsymbol{\omega}) \leq\ & (\|\boldsymbol{s}_1\| + B_{\mathcal{W}}) \|\boldsymbol{b}_t\| \prod_{i=1}^{t-1} (B_{\mathcal{W}} + B_{\mathcal{W}} B_{\mathbb{C}}) B_{\mathcal{W}} \\ & + (\|\boldsymbol{s}_1\| + B_{\mathcal{W}}) \|\boldsymbol{b}_t\| \left\| \sum_{t'=t}^{H} \left(\prod_{i=1;i\neq t}^{t'} (B_{\mathcal{W}} + B_{\mathcal{W}} B_{\mathbb{C}}) \right) B_{\mathcal{W}} \right\|_F . \end{aligned}$$

Since this bound has no dependence on $\boldsymbol{w}$, the desired property is obtained.

Step 3: $\mathcal{J}$ is L-smooth.

For any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}$, we have

$$\begin{aligned} \|\nabla_{\boldsymbol{\theta}} J(\boldsymbol{x};\boldsymbol{w}) - \nabla_{\boldsymbol{\theta}} J(\boldsymbol{y};\boldsymbol{w})\| &\leq \left(\sum_{t=1}^{H-1} \left\| \mathrm{vec}\left(\frac{\partial J(\boldsymbol{x};\boldsymbol{w})}{\partial \boldsymbol{\vartheta}_t} \right) - \mathrm{vec}\left(\frac{\partial J(\boldsymbol{y};\boldsymbol{w})}{\partial \boldsymbol{\vartheta}_t} \right) \right\|^2 \right)^{1/2} \\ &= \left(\sum_{t=1}^{H-1} \left\| \frac{\partial J(\boldsymbol{x};\boldsymbol{w})}{\partial \boldsymbol{\vartheta}_t} - \frac{\partial J(\boldsymbol{y};\boldsymbol{w})}{\partial \boldsymbol{\vartheta}_t} \right\|_F^2 \right)^{1/2} \\ &\leq \left(\sum_{t=1}^{H-1} L_t^2 \|\boldsymbol{x}_t - \boldsymbol{y}_t\|_F^2 \right)^{1/2} \\ &\leq \max_{t\in\mathcal{T}} \{L_t\} \left(\sum_{t=1}^{H-1} \|\boldsymbol{x}_t - \boldsymbol{y}_t\|_F^2 \right)^{1/2} \\ &= \max_{t\in\mathcal{T}} \{L_t\} \|\boldsymbol{x} - \boldsymbol{y}\|. \end{aligned} \tag{35}$$

Consequently, $\|\nabla_{\boldsymbol{\theta}} J(\boldsymbol{x};\boldsymbol{w}) - \nabla_{\boldsymbol{\theta}} J(\boldsymbol{y};\boldsymbol{w})\| \leq L\|\boldsymbol{x} - \boldsymbol{y}\|$, where $L = \max_{t\in\mathcal{T}} \{L_t(\boldsymbol{w})\}$.

Proceeding along similar lines, one can straightforwardly show the boundedness of the gradient

$$\|\nabla_{\boldsymbol{\theta}} J(\boldsymbol{x},\boldsymbol{\omega})\| = \left(\sum_{t=1}^{H} \left\| \mathrm{vec}\left(\frac{\partial J(\boldsymbol{x},\boldsymbol{\omega})}{\partial \boldsymbol{\vartheta}_t} \right) \right\|^2 \right)^{1/2} = \left(\sum_{t=1}^{H} \left\| \frac{\partial J(\boldsymbol{x},\boldsymbol{\omega})}{\partial \boldsymbol{\vartheta}_t} \right\|_F^2 \right)^{1/2}$$

for any $\boldsymbol{x} \in \mathbb{C}$, where

$$\begin{aligned}\left\|\frac{\partial J(\boldsymbol{x};\boldsymbol{w})}{\partial \boldsymbol{\vartheta}_t}\right\|_F &= \left\|\mathbf{R}_{1,t}(\boldsymbol{x})(\boldsymbol{s}_1-\boldsymbol{\mu}_1^s))\boldsymbol{b}_t^\top + \sum_{t'=t}^{H}(\boldsymbol{s}_1-\boldsymbol{\mu}_1^s)(\boldsymbol{b}_t^\top \boldsymbol{x}_t^\top + \boldsymbol{c}_t^\top)\frac{\partial \mathbf{R}_{1,t'}(\boldsymbol{x})}{\partial \boldsymbol{\vartheta}_t}\right\|_F \\ &\le \prod_{i=1}^{t-1}\left(B_{\mathcal{W}}+B_{\mathcal{W}}B_{\mathbb{C}}\right)\left(\|\boldsymbol{s}_1\|+B_{\mathcal{W}}\right)\|\boldsymbol{b}_t\| \\ &\quad + \sum_{t'=t}^{H}(\|\boldsymbol{s}_1\|+B_{\mathcal{W}})(\|\boldsymbol{b}_t\|B_{\mathbb{C}}+\|\boldsymbol{c}_t\|)\left(\prod_{i=1;i\neq t}^{t'}(B_{\mathcal{W}}+B_{\mathcal{W}}B_{\mathbb{C}})\right)B_{\mathcal{W}}.\end{aligned}$$

Thus by the bounded convergence theorem, we have

$$\nabla_{\boldsymbol{\theta}}\mathbb{E}_{\boldsymbol{w}}[J(\boldsymbol{y};\boldsymbol{w})] = \mathbb{E}_{\boldsymbol{w}}[\nabla_{\boldsymbol{\theta}}J(\boldsymbol{x};\boldsymbol{w})] \tag{36}$$

Then, for any $\boldsymbol{x},\boldsymbol{y} \in \mathbb{C}$, we have

$$\begin{aligned}\|\nabla\mathcal{J}(\boldsymbol{x})-\nabla\mathcal{J}(\boldsymbol{y})\|_F &= \|\nabla_{\boldsymbol{\theta}}\mathbb{E}_{\boldsymbol{w}}[J(\boldsymbol{x};\boldsymbol{w})]-\nabla_{\boldsymbol{\theta}}\mathbb{E}_{\boldsymbol{w}}[J(\boldsymbol{y};\boldsymbol{w})]\| \\ &\le \|\mathbb{E}_{\boldsymbol{w}}[\nabla_{\boldsymbol{\theta}}J(\boldsymbol{x};\boldsymbol{w})]-\mathbb{E}_{\boldsymbol{w}}[\nabla_{\boldsymbol{\theta}}J(\boldsymbol{y};\boldsymbol{w})]\| \quad (37)\\ &= \|\mathbb{E}_{\boldsymbol{w}}[\nabla_{\boldsymbol{\theta}}J(\boldsymbol{x};\boldsymbol{w})-\nabla_{\boldsymbol{\theta}}J(\boldsymbol{y};\boldsymbol{w})]\| \\ &\le \mathbb{E}_{\boldsymbol{w}}\left[\|\nabla_{\boldsymbol{\theta}}J(\boldsymbol{x};\boldsymbol{w})-\nabla_{\boldsymbol{\theta}}J(\boldsymbol{y};\boldsymbol{w})\|\right] \quad (38)\\ &= L\|\boldsymbol{x}-\boldsymbol{y}\| \quad (39)\end{aligned}$$

where (37) follows from (36), (38) is due to Jensen's inequality and the convexity of the norm, and (39) follows from (35).

10.6 Proof of Corollary 1

Let $\boldsymbol{\theta}^\star$ be the unique global optimum. Assumption 3 implies $\nabla\mathcal{J}(\boldsymbol{\theta}^\star)=0$. By Lemma 1,

$$\|\nabla\mathcal{J}(\boldsymbol{x})\| = \|\nabla\mathcal{J}(\boldsymbol{x})-\nabla\mathcal{J}(\boldsymbol{\theta}^\star)\| \le L\|\boldsymbol{x}-\boldsymbol{\theta}^\star\| \le L\cdot\max_{\boldsymbol{y}\in\mathbb{C}}\|\boldsymbol{y}-\boldsymbol{\theta}^\star\|.$$

10.7 Proof of Lemma 2

Step 1: For any $\theta \in \mathbb{C}$ and small enough η, $\boldsymbol{\theta}+\eta\nabla\mathcal{J}(\boldsymbol{\theta}) \in \mathbb{C}$.

Recall that, by Assumption 1, there exists c_0 such that, for all $\boldsymbol{\theta}\in\partial\mathbb{C}$, the update $\boldsymbol{\theta}+\eta\frac{\nabla J(\boldsymbol{\theta};\boldsymbol{w})}{\|\nabla J(\boldsymbol{\theta};\boldsymbol{w})\|} \in \mathbb{C}$ for $\eta \le c_0$.

Denote by

$$\mathbb{U}(c_0) = \left\{\eta\frac{\nabla J(\boldsymbol{\theta}:\boldsymbol{w})}{\|\nabla J(\boldsymbol{\theta};\boldsymbol{w})\|} : \boldsymbol{w}\in\mathcal{W} \text{ and } \eta\le c_0\right\}.$$

Let $B_{\boldsymbol{\theta}}(c_0)$ denote the ball with center $\boldsymbol{\theta}$ and radius c_0. Define the convex cone

$$\mathbb{M} = \{\eta(\boldsymbol{v}-\boldsymbol{\theta}) : \boldsymbol{v}\in B_{\boldsymbol{\theta}}(c_0)\cap\mathbb{C} \text{ and } \eta\ge 0\}$$

Since $\boldsymbol{\theta}+\eta\frac{\nabla J(\boldsymbol{\theta};\boldsymbol{w})}{\|\nabla J(\boldsymbol{\theta};\boldsymbol{w})\|} \in B_{\boldsymbol{\theta}}(c_0)\cap\mathbb{C}$ for $\eta\le c_0$ and for all $\boldsymbol{w}\in\mathcal{W}$, we have $\mathbb{U}(c_0)\subset\mathbb{M}$. If we choose the stepsize as $\eta=\|\nabla J(\boldsymbol{\theta};\boldsymbol{w})\|$, then we can rewrite $\nabla_{\boldsymbol{\theta}}J(\boldsymbol{\theta};\boldsymbol{w}) = \eta\frac{\nabla J(\boldsymbol{\theta};\boldsymbol{w})}{\|\nabla J(\boldsymbol{\theta};\boldsymbol{w})\|} \in \mathbb{M}$. Writing out

$$\nabla\mathcal{J}(\boldsymbol{\theta}) = \mathbb{E}[\nabla_{\boldsymbol{\theta}}J(\boldsymbol{\theta};\boldsymbol{w})] = \int\nabla_{\boldsymbol{\theta}}J(\boldsymbol{\theta};\boldsymbol{w})p(\boldsymbol{w}|\mathcal{D})d\boldsymbol{w} = \int_{\mathcal{W}}\nabla_{\boldsymbol{\theta}}J(\boldsymbol{\theta};\boldsymbol{w})p(\boldsymbol{w}|\mathcal{D})d\boldsymbol{w},$$

the convexity of $\mathbb{M}$ implies $\nabla\mathcal{J}(\boldsymbol{\theta})\in\mathbb{M}$, by generalizing the definition of convex combination to include integrals and probability distributions (Boyd et al. 2004). Moreover, by the definition of a cone, we have $\eta\frac{\nabla\mathcal{J}(\boldsymbol{\theta})}{\|\nabla\mathcal{J}(\boldsymbol{\theta})\|}\in\mathbb{M}$ as well.

Since the norm of the vector $\eta\frac{\nabla\mathcal{J}(\boldsymbol{\theta})}{\|\nabla\mathcal{J}(\boldsymbol{\theta})\|}$ equals η, we have $\eta\frac{\nabla\mathcal{J}(\boldsymbol{\theta})}{\|\nabla\mathcal{J}(\boldsymbol{\theta})\|}\in B_{\mathbf{0}}(c_0)\cap\mathbb{M}$ for $\eta\le c_0$, with $\mathbf{0}$ denoting the origin. By the definition of $\mathbb{M}$,

$$\boldsymbol{\theta}+\eta\frac{\nabla\mathcal{J}(\boldsymbol{\theta})}{\|\nabla\mathcal{J}(\boldsymbol{\theta})\|}\in B_{\boldsymbol{\theta}}(c_0)\cap\mathbb{C}, \qquad \eta\le c_0.$$

Let G be the bound on $\|\nabla \mathcal{J}(\boldsymbol{\theta})\|$ obtained from Corollary 1. Then

$$\boldsymbol{\theta} + \frac{\eta}{G}\nabla \mathcal{J}(\boldsymbol{\theta}) \in B_{\boldsymbol{\theta}}(c_0) \cap \mathbb{C}, \qquad \eta \leq c_0,$$

whence the desired result follows.

Step 2: Update is in $\mathbb{C}$ when $\boldsymbol{\theta}$ is close to the boundary.

By Assumption 1, for any $\boldsymbol{x} \in \partial\mathbb{C}$, there exists a neighborhood $B_{\boldsymbol{x}}(\epsilon_{\boldsymbol{x}})$ with radius $\epsilon_{\boldsymbol{x}} > 0$ such that

$$\boldsymbol{\theta} + \eta\nabla \mathcal{J}(\boldsymbol{\theta}) \in \mathbb{C}, \qquad \boldsymbol{\theta} \in B_{\boldsymbol{x}}(\epsilon_{\boldsymbol{x}}) \cap \mathbb{C}, \quad \eta \leq \frac{c_0}{G}.$$

Define

$$\rho_{\boldsymbol{x}} = \sup\left\{\epsilon : \boldsymbol{\theta} + \eta\nabla \mathcal{J}(\boldsymbol{\theta}) \in \mathbb{C}, \qquad \boldsymbol{\theta} \in B_{\boldsymbol{x}}(\epsilon) \cap \mathbb{C}, \quad \eta \leq \frac{c_0}{G}\right\}. \tag{40}$$

Obviously, $\rho_{\boldsymbol{x}} > \epsilon_{\boldsymbol{x}} > 0$. We wish to show that $\inf_{\boldsymbol{x}} \rho_{\boldsymbol{x}} > 0$.

We proceed by contradiction. Suppose that $\inf_{\boldsymbol{x}} \rho_{\boldsymbol{x}} = 0$. Then, there must exist $\boldsymbol{x} \in \partial\mathbb{C}$ and a sequence $\{\boldsymbol{x}_n\}_{n=1}^{\infty} \in \partial\mathbb{C}$ satisfying $\boldsymbol{x}_n \to \boldsymbol{x}$, such that $\rho_{\boldsymbol{x}_n} \to 0$ as $n \to \infty$. At the same time, we must also have $\rho_{\boldsymbol{x}} > 0$. For all sufficiently large n, we have $\|\boldsymbol{x}_n - \boldsymbol{x}\| < \frac{1}{2}\rho_{\boldsymbol{x}}$ and at the same time $\rho_{\boldsymbol{x}_n} < \frac{1}{2}\rho_{\boldsymbol{x}}$. For any such n, take any $\boldsymbol{\theta} \in B_{\boldsymbol{x}_n}(\frac{1}{2}\rho_{\boldsymbol{x}})$. Then, by the triangle inequality, we obtain

$$\|\boldsymbol{\theta} - \boldsymbol{x}\| \leq \|\boldsymbol{\theta} - \boldsymbol{x}_n\| + \|\boldsymbol{x}_n - \boldsymbol{x}\| < \rho_{\boldsymbol{x}}.$$

Therefore, $\boldsymbol{\theta} \in B_{\boldsymbol{x}}(\rho_x)$, and by the definition of $\rho_{\boldsymbol{x}}$ in (40) we have $\theta + \eta\nabla \mathcal{J}(\boldsymbol{\theta}) \in \mathbb{C}$ for all $\eta \leq \frac{c_0}{G}$. But, by the same definition of $\rho_{\boldsymbol{x}_n}$, it follows that $B_{\boldsymbol{x}_n}(\frac{1}{2}\rho_{\boldsymbol{x}}) \subseteq B_{\boldsymbol{x}_n}(\rho_{\boldsymbol{x}_n})$, which contradicts the fact that $\rho_{\boldsymbol{x}_n} < \frac{1}{2}\rho_{\boldsymbol{x}}$. We conclude that $\inf_{\boldsymbol{x}} \rho_{\boldsymbol{x}} > 0$.

Step 3: Final result.

Let $\rho_{\min} = \inf_{\boldsymbol{x}} \rho_{\boldsymbol{x}}$ and suppose that $\boldsymbol{\theta}_k \in B_{\boldsymbol{x}}(\rho_{\min}) \cap \mathbb{C}$. We wish to show that the event of the updated parameters moving a distance more than $\frac{c_0}{G}$ in the direction of $\nabla\widehat{\mathcal{J}}(\boldsymbol{\pi}_{\boldsymbol{\theta}})$ can occur at most finitely many times. That is,

$$\sum_k 1_{\left\{\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\| > \frac{c_0}{\eta_k G}\right\}} < \infty \tag{41}$$

almost surely. To show this, we derive

$$\begin{aligned}
\sum_{k=1}^{\infty} P\left(\left\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right\| > \frac{c_0}{\eta_k G}\right) &\leq \sum_{k=1}^{\infty} \frac{\mathbb{E}\left[\left\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right\|^2\right]}{\frac{c_0^2}{\eta_k^2 G^2}} && (42)\\
&\leq \sum_{k=1}^{\infty} \eta_k^2 \frac{G^2\mathbb{E}\left[\left(2\max\left\{\left\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \nabla\mathcal{J}(\boldsymbol{\theta}_k)\right\|, \|\nabla\mathcal{J}(\boldsymbol{\theta}_k)\|\right\}\right)^2\right]}{c_0^2} && (43)\\
&= 4G^2\sum_{k=1}^{\infty} \eta_k^2 \frac{\mathbb{E}\left[\left\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \nabla\mathcal{J}(\boldsymbol{\theta}_k)\right\|^2 + \|\nabla\mathcal{J}(\boldsymbol{\theta}_k)\|^2\right]}{c_0^2}\\
&\leq 4G^2\sum_{k=1}^{\infty} \eta_k^2 \frac{\mathbb{E}\left[\left\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \nabla\mathcal{J}(\boldsymbol{\theta}_k)\right\|^2\right] + \mathbb{E}\left[\|\nabla\mathcal{J}(\boldsymbol{\theta}_k)\|^2\right]}{c_0^2}\\
&\leq \frac{4G^2(\sigma^2 + G^2)}{c_0^2}\sum_{k=1}^{\infty} \eta_k^2 && (44)\\
&< \infty,
\end{aligned}$$

where (42) is due to Markov's inequality, (43) follows from the triangle inequality, and (44) holds due to Assumption 2 and Corollary 1. We then obtain (41) by the Borel-Cantelli lemma.

In a very similar manner, we can show $\sum_k 1_{\left\{\|\nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\| > \frac{\rho_{\min}}{\eta_k}\right\}} < \infty$ by deriving

$$\begin{aligned}
\sum_{k=1}^{\infty} P\left(\left\|\nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right\| > \frac{\rho_{\min}}{\eta_k}\right) &\leq \sum_{k=1}^{\infty} \frac{\mathbb{E}\left[\left\|\nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right\|^2\right]}{\rho_{\min}^2/\eta_k^2} &(45)\\
&\leq \sum_{k=1}^{\infty} \eta_k^2 \frac{\mathbb{E}\left[\left(2\max\left\{\left\|\nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \nabla \mathcal{J}(\boldsymbol{\theta}_k)\right\|, \|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|\right\}\right)^2\right]}{\rho_{min}^2} &(46)\\
&= 4\sum_{k=1}^{\infty} \eta_k^2 \frac{\mathbb{E}\left[\left\|\nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \nabla \mathcal{J}(\boldsymbol{\theta}_k)\right\|^2 + \|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right]}{\rho_{min}^2}\\
&\leq 4\sum_{k=1}^{\infty} \eta_k^2 \frac{\mathbb{E}\left[\left\|\nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \nabla \mathcal{J}(\boldsymbol{\theta}_k)\right\|^2\right] + \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right]}{\rho_{min}^2}\\
&\leq \frac{4(\sigma^2+G^2)}{\rho_{min}^2}\sum_{k=1}^{\infty}\eta_k^2 &(47)\\
&< \infty,
\end{aligned}$$

where, again, (45) is due to Markov's inequality, (46) follows from the triangle inequality, and (47) holds due to Assumption 2 and Corollary 1.

Thus, we have

$$\lim_{k\to\infty} 1_{\{\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) \notin \mathbb{C}\}} = 0 \text{ a.s.},$$

that is, the updated parameters will always fall within the feasible region for large enough k. Applying the dominated convergence theorem, we obtain $p\left(A_k^c\right) \to 0$, as required.

10.8 Proof of Theorem 2

The proof uses two technical lemmas, which are stated below and proved in separate subsections of this Appendix.

Lemma 3 *For any $\boldsymbol{\theta} \in \mathbb{C}$, we have $\mathbb{E}[\|\widehat{g}_c(\boldsymbol{\theta})\|^2] \leq 4\|\nabla \mathcal{J}(\boldsymbol{\theta})\|^2 + 4\sigma^2$.*

Lemma 4 *At iteration k, if $\boldsymbol{\theta}_k \in \mathbb{C}$ and $\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) \notin \mathbb{C}$, we have the following inequality,*

$$\mathbb{E}[\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \eta_k \widehat{g}_c(\boldsymbol{\theta}_k)\rangle] \geq -\eta_k(G^2 + G\sigma)\left(1 - p(A_k)\right)^{1/2} + \eta_k \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right]$$

Let $\mathcal{J}^\star = \mathcal{J}(\boldsymbol{\theta}^\star)$, and recall the definition

$$\widehat{g}_c(\boldsymbol{\theta}_k) = \frac{1}{\eta_k}\left(\Pi_C\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right) - \boldsymbol{\theta}_k\right).$$

By the property (18), which follows from Lemma 1, we obtain

$$\mathcal{J}(\boldsymbol{\theta}_k) - \mathcal{J}(\boldsymbol{\theta}_{k+1}) \leq -\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \eta_k \widehat{g}_c(\boldsymbol{\theta}_k)\rangle + L\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\|^2. \tag{48}$$

We will take the expectation of both sides of (48). By Lemma 4, we have

$$-\mathbb{E}\left[\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \eta_k \widehat{g}_c(\boldsymbol{\theta}_k)\rangle\right] \leq \eta_k\left(G^2 + G\sigma\right)\left(1 - p(A_k)\right)^{1/2} - \eta_k \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right].$$

Consequently, (48) leads to

$$\begin{aligned}
\mathbb{E}\left[\mathcal{J}(\boldsymbol{\theta}_k)\right] - \mathbb{E}\left[\mathcal{J}(\boldsymbol{\theta}_{k+1})\right] &\leq -\eta_k \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] + \eta_k(G^2 + G\sigma)\left(1 - p(A_k)\right)^{1/2}\\
&\quad + L\eta_k^2 \mathbb{E}\left[\|\widehat{g}_c(\boldsymbol{\theta}_k)\|^2\right]. &(49)
\end{aligned}$$

Rearranging both sides of (49) gives

$$\begin{aligned}
\eta_k \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] &\leq \left(\mathbb{E}\left[\mathcal{J}(\boldsymbol{\theta}_{k+1})\right] - \mathbb{E}\left[\mathcal{J}(\boldsymbol{\theta}_k)\right]\right) + \eta_k(G^2 + G\sigma)\left(1 - p(A_k)\right)^{1/2}\\
&\quad + L\eta_k^2 \mathbb{E}\left[\|\widehat{g}_c(\boldsymbol{\theta}_k)\|^2\right]. &(50)
\end{aligned}$$

By applying Lemma 3, we can rewrite (50) as

$$\begin{aligned}\eta_k \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] &\leq \left(\mathbb{E}\left[\mathcal{J}(\boldsymbol{\theta}_{k+1})\right] - \mathbb{E}\left[\mathcal{J}(\boldsymbol{\theta}_k)\right]\right) + \eta_k(G^2 + G\sigma)\left(1 - p(A_k)\right)^{1/2} \\ &\quad + 4L\eta_k^2 \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] + 4L\eta_k^2\sigma^2 \end{aligned} \tag{51}$$

By moving $4L\eta_k^2 \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right]$ to the left-hand side of (51), we have

$$\eta_k\left(1 - 4\eta_k L\right)\mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] \leq \left(\mathbb{E}\left[\mathcal{J}(\boldsymbol{\theta}_{k+1})\right] - \mathbb{E}\left[\mathcal{J}(\boldsymbol{\theta}_k)\right]\right) + \eta_k(G^2 + G\sigma)\left(1 - p(A_k)\right)^{1/2} + 4L\eta_k^2\sigma^2.$$

When η_k small enough that $1 - 4\eta_k L \geq \frac{1}{2}$, or, equivalently, $\eta_k \leq \frac{1}{8L}$, we obtain the simplified inequality

$$\eta_k\left(1 - 4\eta_k L\right)\mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] \geq \frac{\eta_k}{2}\mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right].$$

Thus, we have

$$\begin{aligned}\frac{1}{2}\mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] &\leq \left(\frac{1}{\eta_k}\mathbb{E}\left[\mathcal{J}(\boldsymbol{\theta}_{k+1})\right] - \frac{1}{\eta_k}\mathbb{E}\left[\mathcal{J}(\boldsymbol{\theta}_k)\right]\right) \\ &\quad + \left(G^2 + G\sigma\right)\left(1 - p(A_k)\right)^{1/2} + 4L\eta_k\sigma^2. \end{aligned} \tag{52}$$

Then, by applying (52) and $\mathcal{J}(\boldsymbol{\theta}_k) \leq \mathcal{J}^\star$, and summing over $k = 1, 2, \ldots, K$, we obtain

$$\begin{aligned}&\frac{1}{2}\sum_{k=1}^{K}\mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] \\ \leq\ & -\frac{1}{\eta_1}\mathbb{E}\left[\mathcal{J}(\boldsymbol{\theta}_1)\right] + \frac{1}{\eta_K}\mathbb{E}\left[\mathcal{J}(\boldsymbol{\theta}_{K+1})\right] + \sum_{k=2}^{K}\left(\frac{1}{\eta_{k-1}} - \frac{1}{\eta_k}\right)\mathbb{E}\left[\mathcal{J}(\boldsymbol{\theta}_k)\right] \\ &+(G^2 + G\sigma)\sum_{k=1}^{K}(1 - p(A_k))^{1/2} + 4L\sigma^2\sum_{k=1}^{K}\eta_k \\ \leq\ & -\frac{1}{\eta_1}\mathcal{J}(\boldsymbol{\theta}_1) + \frac{1}{\eta_K}\mathcal{J}^\star + \sum_{k=2}^{K}\left(\frac{1}{\eta_{k-1}} - \frac{1}{\eta_k}\right)\mathcal{J}^\star + (G^2 + G\sigma)\sum_{k=1}^{K}(1 - p(A_k))^{1/2} + 4L\sigma^2\sum_{k=1}^{K}\eta_k \\ \leq\ & \frac{1}{\eta_1}\left(\mathcal{J}^\star - \mathcal{J}(\boldsymbol{\theta}_1)\right) + (G^2 + G\sigma)\sum_{k=1}^{K}(1 - p(A_k))^{1/2} + 4L\sigma^2\sum_{k=1}^{K}\eta_k. \end{aligned}$$

Multiplying through by $\frac{2}{K}$ gives the result stated in Theorem 2.

10.9 Proof of Lemma 3

To simplify the notation, let $\nabla \mathcal{J}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta})$. Observe that

$$\mathbb{E}[\|\nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}) - \nabla \mathcal{J}(\boldsymbol{\theta})\|^2]^{1/2} \leq \mathbb{E}[\|\nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}) - \nabla \mathcal{J}(\boldsymbol{\theta})\|^4]^{1/4} \leq \sigma, \tag{53}$$

where the first inequality is obtained by applying Jensen's inequality to the convex function $x \mapsto x^2$, and the second inequality follows by Assumption 2.

We then expand $\widehat{g}_c(\boldsymbol{\theta})$ and derive

$$\begin{aligned}
\mathbb{E}\left[\|\widehat{g}_c(\boldsymbol{\theta})\|^2\right] &= \mathbb{E}\left[\left\|\frac{1}{\eta}\left(\Pi_{\mathbb{C}}\left(\boldsymbol{\theta}+\eta\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta})\right)-\boldsymbol{\theta}\right)\right\|^2\right] \\
&= \mathbb{E}\left[\frac{1}{\eta^2}\left\|\Pi_{\mathbb{C}}\left(\boldsymbol{\theta}+\eta\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta})\right)-\Pi_{\mathbb{C}}(\boldsymbol{\theta})\right\|^2\right] \\
&\leq \mathbb{E}\left[\frac{1}{\eta^2}\eta^2\left\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta})\right\|^2\right] \qquad (54) \\
&= \mathbb{E}\left[\left\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta})-\nabla\mathcal{J}(\boldsymbol{\theta})+\nabla\mathcal{J}(\boldsymbol{\theta})\right\|^2\right] \\
&\leq \mathbb{E}\left[\left(\left\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta})-\nabla\mathcal{J}(\boldsymbol{\theta})\right\|+\|\nabla\mathcal{J}(\boldsymbol{\theta})\|\right)^2\right] \\
&\leq \mathbb{E}\left[\left(2\max\left\{\left\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta})-\nabla\mathcal{J}(\boldsymbol{\theta})\right\|,\|\nabla\mathcal{J}(\boldsymbol{\theta})\|\right\}\right)^2\right] \\
&\leq 4\mathbb{E}\left[\max\left\{\left\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta})-\nabla\mathcal{J}(\boldsymbol{\theta})\right\|^2,\|\nabla\mathcal{J}(\boldsymbol{\theta})\|^2\right\}\right] \\
&\leq 4\mathbb{E}\left[\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta})-\nabla\mathcal{J}(\boldsymbol{\theta})\|^2+\|\nabla\mathcal{J}(\boldsymbol{\theta})\|^2\right] \\
&\leq 4\mathbb{E}\left[\|\nabla\mathcal{J}(\boldsymbol{\theta})\|^2\right]+4\sigma^2 \qquad (55)
\end{aligned}$$

where (54) follows from the contraction property of the projection operator, and (55) is obtained by applying (53).

10.10 Proof of Lemma 4

First, we derive

$$\begin{aligned}
\left(\mathbb{E}\left[\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\|^4\right]\right)^{1/4} &= \left(\mathbb{E}\left[\left\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta}_k)-\nabla\mathcal{J}(\boldsymbol{\theta}_k)+\nabla\mathcal{J}(\boldsymbol{\theta}_k)\right\|^4\right]\right)^{1/4} \\
&\leq \left(\mathbb{E}\left[\left\|\nabla\widehat{\mathcal{J}}(\boldsymbol{\theta}_k)-\nabla\mathcal{J}(\boldsymbol{\theta}_k)\right\|^4\right]\right)^{1/4}+\left(\mathbb{E}\left[\|\nabla\mathcal{J}(\boldsymbol{\theta}_k)\|^4\right]\right)^{1/4} \qquad (56) \\
&\leq \left(\mathbb{E}\left[\|\nabla\mathcal{J}(\boldsymbol{\theta}_k)\|^4\right]\right)^{1/4}+\sigma \qquad (57)
\end{aligned}$$

where (56) follows from Minkowski's inequality, and (57) is due to Assumption 2.

We use the following additional notation. Let $\mathcal{F}_k = \sigma(\boldsymbol{\theta}_1, \boldsymbol{w}_1, \ldots, \boldsymbol{w}_{k-1})$ represent the σ-algebra generated by $\boldsymbol{\theta}_1, \boldsymbol{w}_1, \ldots, \boldsymbol{w}_{k-1}$, where each $\boldsymbol{w}_i = \left\{\boldsymbol{w}_i^{(b)}\right\}_{b=1}^{B}$. Note that $\mathcal{F}_k$ is based on the process model parameters collected through the $(k-1)$th iteration, while $\widehat{\mathcal{J}}(\boldsymbol{\theta}_k)$ is estimated by using the new process models $\boldsymbol{w}_k$ obtained in the kth iteration.

In the following, we use the form (17) of the update. We first consider the conditional expectation with respect to $\mathcal{F}_k$, and calculate

$$
\begin{aligned}
& \mathbb{E}\left[\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \eta_k \widehat{g}_c(\boldsymbol{\theta}_k)\rangle \,|\mathcal{F}_k\right] \\
= \;& \mathbb{E}\left[\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \eta_k \widehat{g}_c(\boldsymbol{\theta}_k) - \eta_k \nabla \mathcal{J}(\boldsymbol{\theta}_k)\rangle \,|\mathcal{F}_k\right] + \eta_k \|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2 \\
= \;& \mathbb{E}\left[\left\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right) - \boldsymbol{\theta}_k - \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \eta_k \nabla \mathcal{J}(\boldsymbol{\theta}_k)\right\rangle \Big| \mathcal{F}_k\right] + \eta_k \|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2 \\
= \;& \mathbb{E}\left[\left\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right) - \boldsymbol{\theta}_k - \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right\rangle \Big| \mathcal{F}_k\right] \\
& + \mathbb{E}\left[\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \eta_k \nabla \mathcal{J}(\boldsymbol{\theta}_k)\rangle \Big| \mathcal{F}_k\right] + \eta_k \|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2 \\
= \;& \mathbb{E}\left[\left\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right) - \boldsymbol{\theta}_k - \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right\rangle \Big| \mathcal{F}_k\right] \\
& + \left\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \mathbb{E}\left[\eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \eta_k \nabla \mathcal{J}(\boldsymbol{\theta}_k) \Big| \mathcal{F}_k\right]\right\rangle + \eta_k \|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2 \\
= \;& \mathbb{E}\left[\left\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right) - \boldsymbol{\theta}_k - \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right\rangle \Big| \mathcal{F}_k\right] + \eta_k \|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2 \qquad (58) \\
= \;& -\Big\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \mathbb{E}\left[\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right)\right) 1_{A_k} \Big| \mathcal{F}_k\right] \\
& + \mathbb{E}\left[\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right)\right) 1_{A_k^c} \Big| \mathcal{F}_k\right] \Big\rangle + \eta_k \|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2 \\
= \;& -\Big\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \mathbb{E}\left[\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right)\right) 1_{A_k^c} \Big| \mathcal{F}_k\right] \Big\rangle + \eta_k \|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2 \qquad (59) \\
= \;& -\mathbb{E}\left[1_{A_k^c} \Big\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right) \Big\rangle \Big| \mathcal{F}_k\right] + \eta_k \|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2. \qquad (60)
\end{aligned}
$$

Above, (58) follows from the fact that $\mathbb{E}\left[\nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) \Big| \mathcal{F}_k\right] = \nabla \mathcal{J}(\boldsymbol{\theta}_k)$ because $\widehat{\mathcal{J}}$ is a sample average, and (59) holds because

$$
\mathbb{E}\left[\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right)\right) 1_{A_k} \Big| \mathcal{F}_k\right] = 0.
$$

Notice that $1_{A_k^c} = 1_{A_k^c}^2$. Then, taking unconditional expectations of the preceding quantities, (60) yields

$$
\begin{aligned}
& \mathbb{E}\left[\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \eta_k \widehat{g}_c(\boldsymbol{\theta}_k)\rangle\right] \\
= \;& -\mathbb{E}\left[1_{A_k^c} \Big\langle \nabla \mathcal{J}(\boldsymbol{\theta}_k), \boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right) \Big\rangle\right] + \eta_k \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] \\
= \;& -\mathbb{E}\left[\left\langle 1_{A_k^c} \nabla \mathcal{J}(\boldsymbol{\theta}_k), 1_{A_k^c}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k) - \Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right)\right) \right\rangle\right] + \eta_k \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] \\
\geq \;& -\left(\mathbb{E}\left[1_{A_k^c} \|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right]\right)^{1/2} \left(\mathbb{E}\left[1_{A_k^c} \left\|\Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right) - \boldsymbol{\theta}_k - \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right\|^2\right]\right)^{1/2} \\
& + \eta_k \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] \qquad (61) \\
\geq \;& -\mathbb{E}[1_{A_k^c}]^{1/4} \left(\mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^4\right]\right)^{1/4} \mathbb{E}[1_{A_k^c}]^{1/4} \left(\mathbb{E}\left[\left\|\Pi_{\mathbb{C}}\left(\boldsymbol{\theta}_k + \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right) - \boldsymbol{\theta}_k - \eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\right\|^4\right]\right)^{1/4} \\
& + \eta_k \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] \qquad (62) \\
\geq \;& -\mathbb{E}[1_{A_k^c}]^{1/2} \left(\mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^4\right]\right)^{1/4} \left(\mathbb{E}\left[\|\eta_k \nabla \widehat{\mathcal{J}}(\boldsymbol{\theta}_k)\|^4\right]\right)^{1/4} + \eta_k \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] \qquad (63) \\
\geq \;& -\left(1 - p(A_k)\right)^{1/2} \eta_k \left(\mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^4\right]\right)^{1/4} \left(\left(\mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^4\right]\right)^{1/4} + \sigma\right) + \eta_k \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] \qquad (64) \\
\geq \;& -\left(1 - p(A_k)\right)^{1/2} \eta_k (G^2 + G\sigma) + \eta_k \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right] \qquad (65) \\
\geq \;& -\eta_k (G^2 + G\sigma)\left(1 - p(A_k)\right)^{1/2} + \eta_k \mathbb{E}\left[\|\nabla \mathcal{J}(\boldsymbol{\theta}_k)\|^2\right].
\end{aligned}
$$

Above, (61) and (62) follow by applying Holder's inequality. Then, the inequality (63) follows by applying the second property of the projection in Proposition 4. Notice that $\boldsymbol{\theta}_k \in \mathbb{C}$. The inequality (64) follows from (57), and, finally, (65) is obtained by applying Corollary 1.