

Experiments 3 - Lake 2



Lake problem describes a situation where inhabitants of a town intend to increase their economic benefits through developing industry and agriculture. These activities will also emit pollution into a lake nearby. Although the lake pollution decreases continuously due to natural removal, once it exceeds a threshold X_{crit} , irreversible lake eutrophication would occur and cause huge losses. This problem requires decision-makers to decide the annual pollution emissions, with the aim of maximizing the economic benefit while avoiding eutrophication. The lake pollution transition function is given by:

$$\begin{equation}$$

$$X_{t+1} = X_t + a_t \frac{(X_t^q)}{(1 + X_t^q)} - bX_t + \epsilon_t$$

$$\end{equation}$$

where X_t is the pollution at time t , and $X_0 = 0$; a_t is the rate of anthropogenic pollution at time t ; b is the lake's natural removal rate; q is the lake's natural recycling rate; ϵ_t is the rate of natural pollution at time t . The natural pollution ϵ_t is modeled as a log normal distribution with mean μ and variance σ^2 . The threshold X_{crit} is the pollution level at which the lake's natural recycling equals its natural removal rate. The total benefit from the pollution is given by

$$\begin{equation}$$

$$f_{economic} = \sum_{t \in T} \alpha a_t \delta^t$$

$$\end{equation}$$

where α is the benefit-to-pollution ratio, and δ is the discount rate.

The lake problem is characterized by both stochastic uncertainty and deep uncertainty. The stochastic uncertainty arises from the natural inflow. Deep uncertainty is presented by uncertainty about the mean μ and standard deviation σ of the lognormal distribution characterizing the natural inflow, the natural removal rate of the lake β , the natural recycling rate of the lake q , and the discount rate δ .

Please formalize this problem description as a model. Please also specify key components, such as state variables, stochastic variables, constant parameters, uncertainties and transition functions.

Here's a **formal mathematical model** of the *Lake Problem*, organized by its key components and uncertainty structure.

1. Problem Description

The *Lake Problem* represents a **stochastic dynamic optimization problem** under both **stochastic** and **deep uncertainty**.

A policymaker (decision-maker) controls annual pollution emissions a_t to balance **economic benefits** and **ecological risk**, avoiding irreversible lake eutrophication.

2. Model Structure

Objective Function

$$\max_{\{a_t\}_{t=0}^T} \mathbb{E} \left[\sum_{t=0}^T \alpha a_t \delta^t \right]$$

subject to the lake dynamics and the eutrophication constraint $X_t < X_{\text{crit}}$ for all t .

3. System Dynamics

State Transition Function

$$X_{t+1} = X_t + a_t + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

- X_t : state variable — pollution stock in the lake at time t .
- a_t : decision variable — controllable anthropogenic pollution at time t .
- $\frac{X_t^q}{1+X_t^q}$: nonlinear feedback capturing natural recycling (phosphorus release from sediment).
- bX_t : natural removal process (sedimentation and outflow).
- ϵ_t : stochastic inflow of natural pollution.

4. Key Model Components

Component Type	Symbol	Description
State variable	X_t	Lake pollution stock at time t
Control (decision) variable	a_t	Anthropogenic pollution emission at time t
Stochastic variable	ϵ_t	Natural pollution inflow at time t , lognormally distributed
Threshold (critical state)	X_{crit}	Pollution level beyond which eutrophication is irreversible
Reward (benefit)	$\alpha a_t \delta^t$	Economic benefit from pollution at time t
Discount rate	δ	Time preference factor
Transition parameters	b, q	Natural removal rate and recycling rate
Stochastic parameters	μ, σ	Mean and standard deviation of the lognormal distribution of ϵ_t

5. Stochastic Process

$$\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$$

This captures random variation in natural inflows of pollutants due to rainfall, runoff, etc.

6. Uncertainty Classification

Type of Uncertainty	Uncertain Parameters	Description
Stochastic uncertainty	ϵ_t	Random variability in natural inflow each year
Deep uncertainty	$\mu, \sigma, b, q, \delta$	Unknown or contested system parameters and preferences

Thus, the model must accommodate both **aleatory uncertainty** (randomness) and **epistemic/deep uncertainty** (unknown or ambiguous parameters).

7. Constraints

1. Dynamic constraint:

$$X_{t+1} = X_t + a_t + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

2. Initial condition:

$$X_0 = 0$$

3. Irreversibility constraint:

$$X_t < X_{\text{crit}}, \quad \forall t$$

4. Feasibility of decisions:

$$a_t \geq 0, \quad \forall t$$

8. Extended Formulation for Deep Uncertainty

To represent deep uncertainty, we may treat uncertain parameters as elements of a scenario set:

$$\theta = (\mu, \sigma, b, q, \delta) \in \Theta$$

Decision-making approaches include:

- **Expected utility maximization** (assume probabilistic beliefs over Θ),
- **Robust optimization** (optimize for worst case),
- **Satisficing or adaptive control** (optimize under constraints of performance across scenarios).

9. Full Model Summary

$$\begin{aligned}
 & \max_{\{a_t\}} \quad \mathbb{E}_{\epsilon} \left[\sum_{t=0}^T \alpha a_t \delta^t \right] \\
 & \text{s.t.} \quad X_{t+1} = X_t + a_t + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t, \quad t = 0, 1, \dots, T-1, \\
 & \quad X_0 = 0, \\
 & \quad X_t < X_{\text{crit}}, \quad \forall t, \\
 & \quad a_t \geq 0, \quad \forall t, \\
 & \quad \epsilon_t \sim \text{LogNormal}(\mu, \sigma^2), \\
 & \quad (\mu, \sigma, b, q, \delta) \in \Theta.
 \end{aligned}$$

10. Interpretation

This model captures:

- The **trade-off** between short-term economic gain and long-term ecological sustainability.
- The **risk of crossing a nonlinear environmental threshold**.
- The impact of **stochastic environmental variability** and **structural/deep uncertainty** in system parameters.

Would you like me to also formalize it as a **Markov Decision Process (MDP)** or **stochastic dynamic programming model** (i.e., specifying state, action, transition, and reward functions explicitly for computational formulation)?



Please specify the constant parameters in this problem.

Good question — let's clearly identify the **constant parameters** in the *Lake Problem* model.

These are quantities that are **fixed** within a given model run or scenario (though under *deep uncertainty*, some of them might vary **across scenarios**).

✓ Constant Parameters

Symbol	Parameter Name	Meaning / Role in Model	Typical Notes
α	Benefit-to-pollution ratio	Converts pollution emissions a_t into economic benefits (i.e., the marginal benefit of emitting pollution).	Usually a positive constant ($\alpha > 0$).
X_{crit}	Critical pollution threshold	The pollution level beyond which the lake undergoes irreversible eutrophication.	Fixed environmental constant determined by ecosystem properties.
X_0	Initial pollution stock	The starting pollution level in the lake.	Usually $X_0 = 0$.

Symbol	Parameter Name	Meaning / Role in Model	Typical Notes
T	Time horizon	Number of decision periods (years) considered in the analysis.	Fixed length of simulation or planning horizon.
Θ	Set of possible parameter values	Domain of uncertain parameters (used when modeling deep uncertainty).	Not a stochastic variable — defines possible constants across scenarios.
b	Natural removal rate	Rate at which pollution is removed from the lake via natural processes (sedimentation, outflow).	Treated as constant within a scenario; subject to deep uncertainty across scenarios.
q	Natural recycling rate	Controls the nonlinearity of internal phosphorus feedback (nutrient recycling).	Fixed per scenario; uncertain across scenarios.
μ	Mean of lognormal inflow distribution	Determines expected value of natural inflows ϵ_t .	Constant in a scenario; uncertain across scenarios.
σ	Standard deviation of lognormal inflow distribution	Determines variability of the stochastic inflow ϵ_t .	Constant per scenario; uncertain across scenarios.
δ	Discount rate	Determines time preference (weight of future benefits).	Constant within a scenario; uncertain under deep uncertainty.

Summary by Category

Category	Constants	Comments
Economic constants	α, δ	Determine benefit structure and intertemporal preferences.
Environmental constants	$b, q, X_{\text{crit}}, X_0$	Describe the physical lake system.
Stochastic distribution constants	μ, σ	Define the statistical behavior of random natural inflows.
Modeling constants	T, Θ	Define time horizon and uncertainty set.

In short:

Constant parameters are $\alpha, \delta, b, q, X_{\text{crit}}, X_0, T, \mu, \sigma$.

Some of them ($\mu, \sigma, b, q, \delta$) are *deeply uncertain constants* — fixed within a single model run but uncertain across scenarios.



X_{crit} is not a constant because it depends on other deeply uncertain parameters. Please refine your model specification.

Excellent observation — you're absolutely right.

The critical pollution threshold X_{crit} is **not a fixed constant**, since it **depends on system parameters** that are themselves **deeply uncertain**, specifically the **natural removal rate b** and **natural recycling rate q** .

Let's refine the model specification accordingly.

Refined Model Specification

1. System Dynamics

$$X_{t+1} = X_t + a_t + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

where

- X_t : pollution stock in the lake at time t ,
- a_t : anthropogenic pollution (decision variable),
- b : natural removal rate (deeply uncertain constant),
- q : natural recycling rate (deeply uncertain constant),
- ϵ_t : natural pollution inflow, $\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$.

2. Critical Threshold (Endogenous Relationship)

The **critical pollution threshold** X_{crit} is *defined implicitly* by the point where **natural recycling equals natural removal**:

$$\frac{X_{\text{crit}}^q}{1 + X_{\text{crit}}^q} = bX_{\text{crit}}$$

Thus, X_{crit} is **not constant** — it is an **implicit function** of b and q :

$$X_{\text{crit}} = X_{\text{crit}}(b, q)$$

This means that any change or uncertainty in b or q directly affects the critical threshold.

3. Objective Function

$$\max_{\{a_t\}_{t=0}^T} \mathbb{E} \left[\sum_{t=0}^T \alpha a_t \delta^t \right]$$

subject to the system dynamics and $X_t < X_{\text{crit}}(b, q)$.

4. Model Components (Refined)

Component Type	Symbol	Description	Notes
State variable	X_t	Pollution stock at time t	Dynamic variable
Decision variable	a_t	Anthropogenic pollution emissions	Control variable

Component Type	Symbol	Description	Notes
Stochastic variable	ϵ_t	Natural pollution inflow (lognormal)	$\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$
Derived variable	$X_{\text{crit}}(b, q)$	Critical pollution threshold	Implicitly defined function of uncertain parameters
Reward function	$f_{\text{economic}}(a_t) = \alpha a_t \delta^t$	Economic benefit per time period	
Transition function	$X_{t+1} = X_t + a_t + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t$	Lake dynamics	

5. Constant Parameters (Refined List)

Symbol	Name	Role	Uncertainty Type
α	Benefit-to-pollution ratio	Determines benefit of emissions	Constant
T	Time horizon	Number of decision periods	Constant
X_0	Initial pollution stock	Starting state (often 0)	Constant
μ	Mean of lognormal inflow	Controls stochastic inflow mean	Deep uncertainty
σ	Std. deviation of inflow	Controls inflow variability	Deep uncertainty
b	Natural removal rate	Removal efficiency	Deep uncertainty
q	Natural recycling rate	Recycling nonlinearity	Deep uncertainty
δ	Discount rate	Time preference	Deep uncertainty

So the **true constant parameters** (fixed and known) are:

$$\{\alpha, T, X_0\}$$

The **deeply uncertain parameters** (fixed within scenario, uncertain across scenarios) are:

$$\{\mu, \sigma, b, q, \delta\}$$

and X_{crit} becomes a **derived quantity**:

$$X_{\text{crit}} = X_{\text{crit}}(b, q)$$

6. Complete Refined Model

$$\begin{aligned}
 & \max_{\{a_t\}} \quad \mathbb{E}_\epsilon \left[\sum_{t=0}^T \alpha a_t \delta^t \right] \\
 & \text{s.t.} \quad X_{t+1} = X_t + a_t + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t, \\
 & \quad X_0 = 0, \\
 & \quad X_t < X_{\text{crit}}(b, q), \quad \forall t, \\
 & \quad a_t \geq 0, \quad \forall t, \\
 & \quad \epsilon_t \sim \text{LogNormal}(\mu, \sigma^2), \\
 & \quad (\mu, \sigma, b, q, \delta) \in \Theta.
 \end{aligned}$$

✓ Key Refinement Summary

- X_{crit} **is not a constant**, but **endogenously defined** by b and q .
- The **only true constants** are α, T, X_0 .
- The **deeply uncertain parameters** $(\mu, \sigma, b, q, \delta)$ define scenario-specific behavior.
- $X_{\text{crit}}(b, q)$ links ecosystem dynamics directly to those uncertain parameters.

Would you like me to show how $X_{\text{crit}}(b, q)$ can be solved explicitly (i.e., derive or approximate the analytical form of the threshold)?



Various stakeholders approach this problem from different perspectives. Please refine the model specification from the perspectives of: (1) the local community; and (2) the environmental regulator. These two perspectives should share a common environment, with each stakeholder independently controlling its own control variables and indirectly influencing each other through the shared environment. In the specification, specify the common environment, including global parameters and variables. For each perspective, also specify the corresponding model and its key components, such as control variables, transition functions, and objective functions. If there are multiple objectives, provide multiple objective functions instead of a weighted objective function.

Excellent — this is the next logical refinement.

You're now framing the **Lake Problem** as a **multi-actor system** with **shared environmental dynamics**, where different stakeholders make decisions that interact through the **common lake environment**.

Let's build a **structured, formal multi-agent model** that represents both:

1. **The local community**, which values economic development and benefits from pollution-generating activities; and
2. **The environmental regulator**, which aims to protect the lake and avoid eutrophication.



1. Common Environment (Shared System)

Both stakeholders operate within the **same environmental system**, characterized by the shared **lake pollution stock** X_t .

Global State and Transition Function

$$X_{t+1} = X_t + a_t^{(L)} + a_t^{(R)} + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

where:

- X_t : lake pollution stock (shared **state variable**)
- $a_t^{(L)}$: anthropogenic emissions from **local community** (e.g., industry, agriculture)
- $a_t^{(R)}$: regulatory interventions that **add or remove** pollution (e.g., lake restoration, mitigation investment, or pollution abatement decisions)
- b : natural removal rate (deeply uncertain)
- q : natural recycling rate (deeply uncertain)
- $\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$: stochastic natural inflow
- $X_{\text{crit}}(b, q)$: critical pollution threshold (derived, not constant)

Global Parameters (Common to All Agents)

Symbol	Meaning	Uncertainty Type
b	Natural removal rate	Deep uncertainty
q	Natural recycling rate	Deep uncertainty
μ	Mean of lognormal inflow	Deep uncertainty
σ	Standard deviation of inflow	Deep uncertainty
δ	Discount rate (societal time preference)	Deep uncertainty
X_0	Initial pollution level	Constant
T	Time horizon	Constant
$X_{\text{crit}}(b, q)$	Eutrophication threshold	Derived from b, q

2. Stakeholder Models

(1) Local Community Perspective

The **local community** represents households, farmers, and businesses that emit pollution through productive activities. They benefit economically from higher emissions but suffer indirectly from environmental degradation.

Control Variable

$$a_t^{(L)} \geq 0$$

Anthropogenic pollution (e.g., industrial/agricultural discharge).

State Variable (shared)

$$X_t$$

Transition Function (shared)

$$X_{t+1} = X_t + a_t^{(L)} + a_t^{(R)} + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

Objective Functions

The local community has **two objectives**:

1. Economic Benefit

$$f_1^{(L)} = \sum_{t=0}^T \alpha_L a_t^{(L)} \delta^t$$

- α_L : benefit-to-pollution ratio specific to the community.

2. Environmental Well-being (Disutility from Pollution)

$$f_2^{(L)} = - \sum_{t=0}^T \gamma_L X_t \delta^t$$

- γ_L : sensitivity of community welfare to pollution level (e.g., recreation, health).

The community's decision problem is therefore **multi-objective**:

$$\max_{\{a_t^{(L)}\}} (f_1^{(L)}, f_2^{(L)})$$

subject to the shared dynamics and $X_t < X_{\text{crit}}(b, q)$.

(2) Environmental Regulator Perspective

The **environmental regulator** represents the policymaker or governing agency responsible for maintaining ecological integrity and sustainability.

It may implement pollution controls, clean-up investments, or restoration actions.

Control Variable

$$a_t^{(R)}$$

- Interpreted as **regulatory action** (can be positive or negative):
 - $a_t^{(R)} > 0$: additional pollution (e.g., expansion of infrastructure, allowed permits).

- $a_t^{(R)} < 0$: pollution reduction effort or clean-up.

State Variable (shared)

$$X_t$$

Transition Function (shared)

$$X_{t+1} = X_t + a_t^{(L)} + a_t^{(R)} + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

Objective Functions

The regulator typically balances **ecological** and **social** goals.

1. Ecological Health (Eutrophication Avoidance)

$$f_1^{(R)} = - \sum_{t=0}^T (X_t - X_{\text{crit}}(b, q))_+^2 \delta^t$$

- Penalizes pollution exceeding or approaching the critical threshold.

2. Economic Stability / Policy Cost

$$f_2^{(R)} = - \sum_{t=0}^T c_R (a_t^{(R)})^2 \delta^t$$

- c_R : cost coefficient for regulatory interventions (e.g., abatement or restoration costs).

Hence, the regulator's decision problem is:

$$\max_{\{a_t^{(R)}\}} (f_1^{(R)}, f_2^{(R)})$$

subject to the shared dynamics and $X_t < X_{\text{crit}}(b, q)$.



3. Shared Dynamics and Interactions

Both stakeholders' actions influence the **same environmental state**:

$$X_{t+1} = X_t + a_t^{(L)} + a_t^{(R)} + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

Thus:

- The **local community's emissions** directly increase pollution.
- The **regulator's actions** can increase or offset it.
- Both observe and respond to the **common state** X_t .
- Their goals are **partially conflicting** (economic vs. environmental).



4. Summary Tables

(a) Common Environment

Element	Symbol	Type	Description
State variable	X_t	Shared	Lake pollution stock
Transition function	$X_{t+1} = X_t + a_t^{(L)} + a_t^{(R)} + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t$	Shared	Environmental dynamics
Stochastic variable	ϵ_t	Shared	Natural inflow (lognormal)
Deeply uncertain parameters	$b, q, \mu, \sigma, \delta$	Shared	Physical and economic system uncertainty
Derived threshold	$X_{\text{crit}}(b, q)$	Shared	Eutrophication limit

(b) Local Community Model

Component	Symbol / Function	Description
Control variable	$a_t^{(L)} \geq 0$	Emission rate
Objectives	$f_1^{(L)} = \sum \alpha_L a_t^{(L)} \delta^t; f_2^{(L)} = - \sum \gamma_L X_t \delta^t$	Economic vs. environmental concern
Constraints	Shared environment dynamics; $X_t < X_{\text{crit}}(b, q)$	Environmental limit

(c) Regulator Model

Component	Symbol / Function	Description
Control variable	$a_t^{(R)} \in \mathbb{R}$	Regulatory action (pollution control or restoration)
Objectives	$f_1^{(R)} = - \sum (X_t - X_{\text{crit}})_+^2 \delta^t; f_2^{(R)} = - \sum c_R (a_t^{(R)})^2 \delta^t$	Ecological protection vs. cost efficiency
Constraints	Shared environment dynamics; $X_t < X_{\text{crit}}(b, q)$	Ecological limit



5. Conceptual Interpretation

- The **local community** and **regulator** are **independent decision-makers** coupled through the **shared environmental state** X_t .
- This structure defines a **dynamic game** or **multi-agent system**.
- Coordination or policy instruments (e.g., taxes, subsidies, caps) could be added later to explore cooperative vs. non-cooperative outcomes.
- The shared environment ensures that **one agent's decision** affects the other's **state evolution** and thus both agents' objectives.

Would you like me to **extend this formulation into a dynamic game representation** (e.g., define Nash equilibrium or cooperative Pareto-optimal solutions between the two agents)? That would be the next natural step.



To ensure internal consistency across these models, please compose them into a single, unified model that shares a common environment and states, while embedding perspective-specific controls and objectives. The composed model should: 1. embed each perspective's control variables and objectives; 2. enable modular evaluation of each perspective; 3. for each perspective, treat other perspectives' controls as exogenous inputs; and 4. incorporate all other components from every perspective.

Perfect — this is exactly the next logical refinement.

You now want a **unified, modular, and internally consistent formulation** of the *Lake Problem under multiple stakeholder perspectives* — where both actors (local community and regulator) operate in a **shared environmental system** but have **distinct decision variables** and **separate objectives**.

The unified model must:

1. Share a **common environment** (state, stochastic process, uncertain parameters).
2. Embed **each perspective's control variables** and **objectives**.
3. Allow **modular evaluation** (each actor sees others' actions as exogenous).
4. Maintain full consistency in transitions and parameters.



Unified Multi-Perspective Lake Problem Model

1. Common Environment (Shared Global System)

The **lake pollution stock** X_t is the shared environmental state.

State Transition Function

$$X_{t+1} = X_t + a_t^{(L)} + a_t^{(R)} + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

- X_t : lake pollution stock (global **state variable**)
- $a_t^{(L)}$: **local community control** — pollution emissions
- $a_t^{(R)}$: **regulator control** — pollution abatement or policy action (can be positive or negative)
- b : natural removal rate (deep uncertainty)
- q : natural recycling rate (deep uncertainty)
- $\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$: stochastic natural inflow
- $X_{\text{crit}}(b, q)$: eutrophication threshold derived implicitly from

$$\frac{X_{\text{crit}}^q}{1 + X_{\text{crit}}^q} = bX_{\text{crit}}.$$

Shared Parameters

Category	Symbol	Description	Type
Environmental	b	Natural removal rate	Deep uncertainty
Environmental	q	Natural recycling rate	Deep uncertainty
Stochastic inflow	μ, σ	Parameters of lognormal inflow	Deep uncertainty
Economic	δ	Discount rate	Deep uncertainty
Model	T	Time horizon	Constant
Initial condition	X_0	Initial pollution stock	Constant
Derived threshold	$X_{\text{crit}}(b, q)$	Eutrophication limit	Derived

All agents share and observe X_t and the stochastic realization ϵ_t .

2. Stakeholder Perspectives Embedded in the Unified Model

We now define **two modular subsystems**, each representing a stakeholder with its own controls and objectives, but coupled through the shared state transition.

(a) Local Community Submodel

Control Variable

$$a_t^{(L)} \geq 0$$

Represents emissions from economic activities (industry, agriculture).

State Variable

$$X_t$$

(shared from the global system).

Transition (inherited from global model)

$$X_{t+1} = X_t + a_t^{(L)} + a_t^{(R)} + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

where $a_t^{(R)}$ is treated as **exogenous** by the local community.

Objective Functions

1. Economic Benefit

$$f_1^{(L)} = \sum_{t=0}^T \alpha_L a_t^{(L)} \delta^t$$

with $\alpha_L > 0$: benefit-to-pollution ratio.

2. Environmental Well-being / Pollution Disutility

$$f_2^{(L)} = - \sum_{t=0}^T \gamma_L X_t \delta^t$$

with $\gamma_L > 0$: disutility parameter representing pollution damage (e.g., health, tourism, local ecology).

Local Community Decision Problem

$$\max_{\{a_t^{(L)}\}} (f_1^{(L)}, f_2^{(L)})$$

subject to the shared state transition and $X_t < X_{\text{crit}}(b, q)$.

(b) Environmental Regulator Submodel

Control Variable

$$a_t^{(R)} \in \mathbb{R}$$

Represents pollution control or intervention:

- $a_t^{(R)} < 0$: pollution abatement / restoration
- $a_t^{(R)} > 0$: additional allowance or infrastructure-related pollution

State Variable

$$X_t$$

(shared from the global system).

Transition (inherited from global model)

$$X_{t+1} = X_t + a_t^{(L)} + a_t^{(R)} + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

where $a_t^{(L)}$ is **exogenous** from the regulator's perspective.

Objective Functions

1. Ecological Protection (Eutrophication Avoidance)

$$f_1^{(R)} = - \sum_{t=0}^T (X_t - X_{\text{crit}}(b, q))_+^2 \delta^t$$

Penalizes pollution approaching or exceeding the eutrophication threshold.

2. Policy Cost (Economic Efficiency)

$$f_2^{(R)} = - \sum_{t=0}^T c_R (a_t^{(R)})^2 \delta^t$$

where $c_R > 0$: cost coefficient for regulatory actions.

Regulator Decision Problem

$$\max_{\{a_t^{(R)}\}} (f_1^{(R)}, f_2^{(R)})$$

subject to the shared state transition and $X_t < X_{\text{crit}}(b, q)$.

3. Unified Model Composition

The full model integrates both perspectives through the shared environment:

State transition: $X_{t+1} = X_t + a_t^{(L)} + a_t^{(R)} + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$

Initial condition: $X_0 = 0$

Constraints: $X_t < X_{\text{crit}}(b, q), \quad a_t^{(L)} \geq 0$

Each perspective:

- Observes the **shared state** X_t and stochastic input ϵ_t ,
- Controls its own variable ($a_t^{(L)}$ or $a_t^{(R)}$),
- Treats the other's control as **exogenous input**,
- Evaluates its own **objectives** independently.

Formal Unified Model

Common environment: $X_{t+1} = X_t + a_t^{(L)} + a_t^{(R)} + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$

$$\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2),$$

$$X_{\text{crit}} = X_{\text{crit}}(b, q),$$

$$(\mu, \sigma, b, q, \delta) \in \Theta,$$

$$X_0 = 0, \quad X_t < X_{\text{crit}}(b, q)$$

$$\begin{aligned} \text{Local community module: } & \begin{cases} \max_{\{a_t^{(L)}\}} \left(f_1^{(L)}, f_2^{(L)} \right) \\ f_1^{(L)} = \sum_{t=0}^T \alpha_L a_t^{(L)} \delta^t \\ f_2^{(L)} = - \sum_{t=0}^T \gamma_L X_t \delta^t \end{cases} \\ \text{Regulator module: } & \begin{cases} \max_{\{a_t^{(R)}\}} \left(f_1^{(R)}, f_2^{(R)} \right) \\ f_1^{(R)} = - \sum_{t=0}^T (X_t - X_{\text{crit}}(b, q))^2_+ \delta^t \\ f_2^{(R)} = - \sum_{t=0}^T c_R (a_t^{(R)})^2 \delta^t \end{cases} \end{aligned}$$

4. Model Properties

Feature	Description
Shared state	X_t : both perspectives affect and observe the same environmental variable.
Perspective-specific control	$a_t^{(L)}$ and $a_t^{(R)}$ are distinct but coupled through X_t .
Perspective-specific objectives	Each stakeholder maximizes different objectives reflecting economic vs. environmental priorities.
Modularity	Each model can be evaluated independently by fixing the other's control sequence (exogenous).
Uncertainty	Deep uncertainty over $(\mu, \sigma, b, q, \delta)$ and stochastic variability ϵ_t .
Interdependence	Decisions interact indirectly through the environmental feedback in X_t .

5. Conceptual Extensions (optional)

This unified model provides a foundation for:

- **Dynamic games:** solving for Nash or cooperative equilibria.
- **Scenario discovery / robust decision-making:** exploring policy resilience under deep uncertainty.
- **Multi-objective evaluation:** analyzing trade-offs between economic and ecological outcomes.

In summary:

This **composed unified model** integrates:

- A **common environment** with shared state and uncertainty structure,
- **Perspective-specific controls and objectives**,
- **Independent decision modules** that treat other actors' actions as exogenous,
- And **internal consistency** across all dynamics, variables, and parameters.

Would you like me to formalize this unified framework next as a **dynamic game** (i.e., specify agents, information sets, and equilibrium concepts)?



Now, provide a modular Python implementation of the unified model, using classes to represent different perspectives. At the end of the program, define a function as the interface. This function should accept a dictionary of control variables and a dictionary of uncertain parameter values as input, simulate the model dynamics, and return the value of each objective function as output. This function does not need to be executed.

Here is a clean, **modular Python implementation** of the **unified multi-perspective Lake Problem model**.

It uses **object-oriented design** so that each stakeholder (Local Community and Regulator) is encapsulated in its own class, while both share a common **LakeEnvironment**.

At the end, a single interface function **simulate_lake_problem()** takes control inputs and uncertain parameter values, runs the shared model, and returns each stakeholder's objective values.

```
python

import numpy as np

# =====
# 1. Common Environment Class
# =====

class LakeEnvironment:
    """
    Represents the shared lake pollution environment.
    Handles system dynamics and stochastic inflows.
    """

    def __init__(self, params):
        """
        Initialize the shared environment.

        Parameters:
            params (dict): Global and uncertain parameters.
                Required keys:
                - b: natural removal rate
                - q: recycling rate
                - mu: mean of lognormal inflow
                - sigma: std. dev. of lognormal inflow
                - delta: discount rate
                - T: time horizon
                - X0: initial pollution level
        """
        self.b = params["b"]
        self.q = params["q"]
        self.mu = params["mu"]
        self.sigma = params["sigma"]
        self.delta = params["delta"]
        self.T = params["T"]
        self.X0 = params.get("X0", 0.0)

        # Compute eutrophication threshold X_crit(b, q)
        self.X_crit = self.compute_Xcrit()

    def compute_Xcrit(self):
        """
        Solve for X_crit(b, q) implicitly where recycling = removal:
        X^q / (1 + X^q) = bX
        Uses a numerical search since closed-form is not available.
        """
        X_candidates = np.linspace(1e-6, 10, 10000)
        lhs = X_candidates**self.q / (1 + X_candidates**self.q)
        rhs = self.b * X_candidates
        idx = np.argmin(np.abs(lhs - rhs))
        return X_candidates[idx]
```

```

def stochastic_inflow(self):
    """Draw stochastic natural inflow  $\varepsilon_t$  from LogNormal( $\mu$ ,  $\sigma^2$ )."""
    return np.random.lognormal(mean=self.mu, sigma=self.sigma)

def transition(self, X_t, a_L, a_R, epsilon_t):
    """Compute the next pollution level  $X_{t+1}$ ."""
    return X_t + a_L + a_R + (X_t**self.q) / (1 + X_t**self.q) - self.b * X_t + epsilon_t

# =====
# 2. Local Community Class
# =====

class LocalCommunity:
    """
    Represents the local community perspective.
    Controls pollution emissions for economic benefit.
    """

    def __init__(self, alpha_L, gamma_L):
        """
        Parameters:
            alpha_L: benefit-to-pollution ratio
            gamma_L: disutility coefficient for pollution
        """
        self.alpha_L = alpha_L
        self.gamma_L = gamma_L

    def objectives(self, emissions, states, delta):
        """
        Compute the local community's objectives.

        Returns:
            f1 (float): total discounted economic benefit
            f2 (float): total discounted disutility from pollution
        """
        T = len(emissions)
        discounts = np.array([delta**t for t in range(T)])
        f1 = np.sum(self.alpha_L * np.array(emissions) * discounts)
        f2 = -np.sum(self.gamma_L * np.array(states) * discounts)
        return f1, f2

# =====
# 3. Environmental Regulator Class
# =====

class EnvironmentalRegulator:
    """
    Represents the environmental regulator perspective.
    Controls abatement or restoration actions.
    """

    def __init__(self, c_R):
        """
        Parameters:
            c_R: cost coefficient for regulatory actions
        """
        self.c_R = c_R

    def objectives(self, actions, states, X_crit, delta):
        """
        Compute the regulator's objectives.

        Returns:
            f1 (float): ecological protection (negative eutrophication penalty)
            f2 (float): policy cost (negative cost)
        """
        T = len(actions)
        discounts = np.array([delta**t for t in range(T)])
        penalty = np.maximum(np.array(states) - X_crit, 0.0)
        f1 = -np.sum((penalty**2) * discounts)
        f2 = -np.sum(self.c_R * np.array(actions)**2 * discounts)
        return f1, f2

```

```
# =====
# 4. Unified Simulation Interface
# =====

def simulate_lake_problem(controls, params, random_seed=None):
    """
    Unified interface for simulating the lake problem under multiple perspectives.

    Parameters:
        controls (dict): Control time series for each perspective.
            Required keys:
            - 'local': list or array of local community emissions ( $a_t^L$ )
            - 'regulator': list or array of regulator actions ( $a_t^R$ )
        params (dict): Global and uncertain parameters.
            Required keys:
            - b, q, mu, sigma, delta, T, X0
            - alpha_L, gamma_L, c_R
        random_seed (int, optional): Seed for reproducibility.

    Returns:
        results (dict): Dictionary containing each perspective's objectives.
        {
            'local': {'economic_benefit': f1_L, 'environment_disutility': f2_L},
            'regulator': {'ecological_protection': f1_R, 'policy_cost': f2_R},
            'trajectory': {'X': X_series}
        }
    """
    if random_seed is not None:
        np.random.seed(random_seed)

    # Initialize shared environment
    env = LakeEnvironment(params)

    # Initialize stakeholders
    local = LocalCommunity(params["alpha_L"], params["gamma_L"])
    regulator = EnvironmentalRegulator(params["c_R"])

    # Unpack control variables
    a_L_series = np.array(controls["local"])
    a_R_series = np.array(controls["regulator"])
    T = params["T"]

    # Simulate lake pollution dynamics
    X_series = [env.X0]
    for t in range(T):
        epsilon_t = env.stochastic_inflow()
        X_next = env.transition(X_series[-1], a_L_series[t], a_R_series[t], epsilon_t)
        X_series.append(X_next)

    # Compute objectives for each perspective
    f1_L, f2_L = local.objectives(a_L_series, X_series[:-1], env.delta)
    f1_R, f2_R = regulator.objectives(a_R_series, X_series[:-1], env.X_crit, env.delta)

    # Collect results
    results = {
        "local": {
            "economic_benefit": f1_L,
            "environment_disutility": f2_L
        },
        "regulator": {
            "ecological_protection": f1_R,
            "policy_cost": f2_R
        },
        "trajectory": {
            "X": X_series
        }
    }

    return results

# =====
# Example (not executed):
```

```
# =====
# controls = {
#     'local': [0.5] * 50,
#     'regulator': [-0.1] * 50
# }
# params = {
#     'b': 0.4, 'q': 2.0, 'mu': 0.01, 'sigma': 0.02,
#     'delta': 0.95, 'T': 50, 'X0': 0.0,
#     'alpha_L': 1.0, 'gamma_L': 0.5, 'c_R': 0.1
# }
# results = simulate_lake_problem(controls, params)
# print(results)
```

✓ Design Highlights

Design Feature	Description
Modular architecture	Each stakeholder (community, regulator) and the environment are independent, reusable modules.
Shared state	The LakeEnvironment class maintains the shared pollution stock and dynamics.
Perspective independence	Each stakeholder class has its own objective functions and treats others' actions as exogenous.
Unified interface	simulate_lake_problem() orchestrates the model, taking control and parameter dictionaries as input and returning all objective outcomes.
Reproducibility	Optional random_seed ensures stochastic reproducibility.

Would you like me to extend this model so that each stakeholder can be **simulated interactively** (e.g., best-response updates or Nash equilibrium search)?



There are two areas where your implementation could be improved: 1. The possible value of X_{crit} is in the range [0.01, 1.5], and its exact value can be estimated using the root-finding function; 2. Your implementation of the lognormal distribution is incorrect. Please refine your implementation.

Excellent — those are very precise and important improvements.

Let's address both issues carefully and update the code accordingly:

✓ Corrections

(1) X_{crit} Calculation

- Previously, X_{crit} was approximated by scanning over a wide range (0 to 10).
- Now, we'll **restrict the search domain** to 0.01, 1.5 (the plausible range).
- We'll use a **proper root-finding algorithm** (`scipy.optimize.root_scalar`) to solve:

$$f(X) = \frac{X^q}{1 + X^q} - bX = 0$$

for $X \in [0.01, 1.5]$.

(2) Lognormal Distribution Correction

- Previously, the `np.random.lognormal(mean=self.mu, sigma=self.sigma)` treated μ, σ as **log-space parameters**, but in this model μ and σ are defined as the **mean and standard deviation in real space**, not log-space.
- We'll convert from real-space mean and std (μ, σ) to log-space parameters:

$$\text{shape} = \sqrt{\ln \left(1 + \frac{\sigma^2}{\mu^2} \right)}, \quad \text{scale} = \frac{\mu}{\sqrt{1 + \frac{\sigma^2}{\mu^2}}}$$

Then draw from `np.random.lognormal(mean=np.log(scale), sigma=shape)`.

Refined Modular Python Implementation

python

```
import numpy as np
from scipy.optimize import root_scalar

# =====
# 1. Common Environment Class
# =====

class LakeEnvironment:
    """
    Shared lake environment with stochastic inflows and nonlinear pollution dynamics.
    """

    def __init__(self, params):
        """
        Parameters:
            params (dict): global and uncertain parameters.
            Required keys:
                - b, q, mu, sigma, delta, T, X0
        """
        self.b = params["b"]
        self.q = params["q"]
        self.mu = params["mu"]
        self.sigma = params["sigma"]
        self.delta = params["delta"]
        self.T = params["T"]
        self.X0 = params.get("X0", 0.0)

        # Compute critical pollution threshold X_crit(b, q)
        self.X_crit = self.compute_Xcrit()

    # -----
    # Root-finding for X_crit(b, q)
    # -----
    def compute_Xcrit(self):
        """
        Solve f(X) = X^q/(1 + X^q) - bX = 0 for X in [0.01, 1.5].
        Uses robust root-finding (brentq method).
        """
        def f(X):
            return (X**self.q) / (1 + X**self.q) - self.b * X

        try:
            sol = root_scalar(f, bracket=[0.01, 1.5], method="brentq")
```

```

        if sol.converged:
            return sol.root
        else:
            # fallback: use midpoint if solver fails
            return 0.5 * (0.01 + 1.5)
    except ValueError:
        # fallback: default midpoint if no valid bracket
        return 0.5 * (0.01 + 1.5)

# -----
# Corrected lognormal stochastic inflow
# -----
def stochastic_inflow(self):
    """
    Draw stochastic inflow  $\varepsilon_t$  from LogNormal with real-space mean= $\mu$ , std= $\sigma$ .
    Converts  $\mu$ ,  $\sigma$  to log-space parameters.
    """
    shape = np.sqrt(np.log(1 + (self.sigma / self.mu) ** 2))
    scale = self.mu / np.sqrt(1 + (self.sigma / self.mu) ** 2)
    return np.random.lognormal(mean=np.log(scale), sigma=shape)

# -----
def transition(self, X_t, a_L, a_R, epsilon_t):
    """Compute next pollution level  $X_{t+1}$ ."""
    return X_t + a_L + a_R + (X_t**self.q) / (1 + X_t**self.q) - self.b * X_t + epsilon_t

# =====
# 2. Local Community Class
# =====

class LocalCommunity:
    """Local community perspective: maximize economic benefit, minimize pollution damage."""

    def __init__(self, alpha_L, gamma_L):
        self.alpha_L = alpha_L
        self.gamma_L = gamma_L

    def objectives(self, emissions, states, delta):
        T = len(emissions)
        discounts = np.array([delta**t for t in range(T)])
        f1 = -np.sum(self.alpha_L * np.array(emissions) * discounts)
        f2 = -np.sum(self.gamma_L * np.array(states) * discounts)
        return f1, f2

# =====
# 3. Environmental Regulator Class
# =====

class EnvironmentalRegulator:
    """Regulator perspective: protect ecosystem and minimize policy cost."""

    def __init__(self, c_R):
        self.c_R = c_R

    def objectives(self, actions, states, X_crit, delta):
        T = len(actions)
        discounts = np.array([delta**t for t in range(T)])
        penalty = np.maximum(np.array(states) - X_crit, 0.0)
        f1 = -np.sum((penalty**2) * discounts) # ecological protection
        f2 = -np.sum(self.c_R * np.array(actions) ** 2 * discounts) # policy cost
        return f1, f2

# =====
# 4. Unified Simulation Interface
# =====

def simulate_lake_problem(controls, params, random_seed=None):
    """
    Unified simulation of the multi-perspective lake problem.

    Parameters:
        controls (dict): Control time series.
    """

```

```

- 'local': list/array of local emissions a_t^(L)
- 'regulator': list/array of regulator actions a_t^(R)
params (dict): Model parameters (including uncertainties).
Required keys:
    b, q, mu, sigma, delta, T, X0, alpha_L, gamma_L, c_R
random_seed (int, optional): Seed for reproducibility.

Returns:
dict: {
    'local': {'economic_benefit': f1_L, 'environment_disutility': f2_L},
    'regulator': {'ecological_protection': f1_R, 'policy_cost': f2_R},
    'trajectory': {'X': X_series, 'Xcrit': X_crit}
}
"""
if random_seed is not None:
    np.random.seed(random_seed)

# Initialize shared environment
env = LakeEnvironment(params)

# Initialize agents
local = LocalCommunity(params["alpha_L"], params["gamma_L"])
regulator = EnvironmentalRegulator(params["c_R"])

# Unpack controls
a_L_series = np.array(controls["local"])
a_R_series = np.array(controls["regulator"])
T = params["T"]

# Simulate pollution trajectory
X_series = [env.X0]
for t in range(T):
    epsilon_t = env.stochastic_inflow()
    X_next = env.transition(X_series[-1], a_L_series[t], a_R_series[t], epsilon_t)
    X_series.append(X_next)

# Compute objectives
f1_L, f2_L = local.objectives(a_L_series, X_series[:-1], env.delta)
f1_R, f2_R = regulator.objectives(a_R_series, X_series[:-1], env.X_crit, env.delta)

return {
    "local": {
        "economic_benefit": f1_L,
        "environment_disutility": f2_L
    },
    "regulator": {
        "ecological_protection": f1_R,
        "policy_cost": f2_R
    },
    "trajectory": {
        "X": X_series,
        "Xcrit": env.X_crit
    }
}

# =====
# Example usage (not executed):
# =====
# controls = {
#     'local': [0.5] * 50,
#     'regulator': [-0.1] * 50
# }
# params = {
#     'b': 0.4, 'q': 2.0, 'mu': 0.01, 'sigma': 0.002,
#     'delta': 0.95, 'T': 50, 'X0': 0.0,
#     'alpha_L': 1.0, 'gamma_L': 0.5, 'c_R': 0.1
# }
# results = simulate_lake_problem(controls, params, random_seed=42)
# print(results)

```

✓ Improvements Summary

Issue	Fix
X_{crit} calculation	Replaced brute-force search with root_scalar (Brent's method), constrained to 0.01, 1.5 .
Lognormal inflow	Corrected to interpret μ, σ as real-space mean/std; converted to log-space using standard transformations.
Numerical robustness	Added fallback for root_scalar failure cases.
Output clarity	Returned both X trajectory and Xcrit in results.

Would you like me to extend this version so it can run *multiple uncertainty scenarios* (e.g., Monte Carlo evaluation over different uncertain parameters)?