



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;  $\epsilon_t$  is the rate of natural pollution at time  $t$ . The natural pollution  $\epsilon_t$  is modeled as a log normal distribution with mean  $\mu$  and variance  $\sigma^2$ . 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  $\alpha$  is the benefit-to-pollution ratio, and  $\delta$  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  $\mu$  and standard deviation  $\sigma$  of the lognormal distribution characterizing the natural inflow, the natural removal rate of the lake  $\beta$ , the natural recycling rate of the lake  $q$ , and the discount rate  $\delta$ .

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*, structured to clearly specify all its components — including states, decisions, uncertainties, and the transition dynamics.

## 1. Model Overview

The **Lake Problem** is a **stochastic dynamic optimization** problem, where the goal is to determine the optimal pollution emissions  $a_t$  over time to **maximize discounted economic benefits** while **avoiding irreversible eutrophication** of a lake.

## 2. Time Horizon

Let

$$t = 0, 1, 2, \dots, T$$

represent discrete time steps (years).

---

### 3. State Variable

- $X_t$ : level of pollution (phosphorus concentration) in the lake at time  $t$ .
    - **State space:**  $X_t \in [0, X_{\max}]$
    - **Initial condition:**  $X_0 = 0$
- 

### 4. Decision (Control) Variable

- $a_t$ : anthropogenic pollution inflow (decision variable).
  - **Control space:**  $a_t \in [0, a_{\max}]$

This represents human-controlled pollution emissions from industry or agriculture.

---

### 5. Stochastic Variable

- $\epsilon_t$ : natural pollution inflow (random shock).

Modeled as:

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

where  $\mu$  and  $\sigma$  represent the mean and standard deviation of the natural inflow.

---

### 6. Transition Function (Lake Pollution Dynamics)

The evolution of lake pollution is given by:

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

where:

- $b$ : natural removal rate of the lake,
  - $q$ : natural recycling rate of the lake,
  - $\frac{X_t^q}{1 + X_t^q}$ : represents feedback recycling of phosphorus from the sediment.
- 

### 7. Objective Function (Economic Benefit)

The total expected discounted benefit is:

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

where:

- $\alpha$ : benefit-to-pollution ratio (economic gain per unit of emission),
- $\delta$ : discount rate,
- $\mathbb{E}[\cdot]$ : expectation over stochastic inflows  $\epsilon_t$ .

## 8. Constraint (Eutrophication Threshold)

To prevent irreversible eutrophication, pollution must not exceed the critical level:

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

where  $X_{\text{crit}}$  is defined implicitly as the pollution level satisfying:

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

(i.e., where natural recycling equals natural removal).

## 9. Uncertainty Characterization

The model includes two types of uncertainty:

### (a) Stochastic uncertainty (short-term randomness)

- Arises from  $\epsilon_t$ , the natural inflow.
- Modeled via the lognormal distribution  $\text{Lognormal}(\mu, \sigma^2)$ .

### (b) Deep uncertainty (parameter uncertainty)

- Arises from uncertain or contested parameters:
  - $\mu, \sigma$ : parameters of the lognormal inflow distribution,
  - $b$ : natural removal rate of the lake,
  - $q$ : recycling rate of the lake,
  - $\delta$ : discount rate.

These can be represented as **uncertain parameters** rather than random variables with known distributions, i.e.:

$$(\mu, \sigma, b, q, \delta) \in \mathcal{U}$$

where  $\mathcal{U}$  is the uncertainty set defined by plausible ranges.

## 10. Model Summary Table

Component	Symbol	Description	Type
State variable	$X_t$	Lake pollution level	Endogenous
Control variable	$a_t$	Anthropogenic pollution emissions	Decision
Stochastic variable	$\epsilon_t$	Natural inflow (lognormal)	Random
Constant parameters	$\alpha, X_0, X_{\text{crit}}$	Benefit ratio, initial state, critical threshold	Known
Uncertain parameters	$\mu, \sigma, b, q, \delta$	Distribution & lake process parameters	Deep uncertainty
Transition function	$X_{t+1} = X_t + a_t + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t$	System dynamics	Deterministic + stochastic
Objective	$\max \mathbb{E} [\sum \alpha a_t \delta^t]$	Maximize discounted benefit	Optimization target
Constraint	$X_t \leq X_{\text{crit}}$	Avoid eutrophication	Safety constraint

## 11. Problem Formulation (Compact Form)

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

Would you like me to extend this into a **robust optimization** or **stochastic dynamic programming** formulation (e.g., Bellman equation form), depending on whether you want to handle deep uncertainty explicitly?



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

Excellent point — you are correct.

Because the **critical pollution threshold**  $X_{\text{crit}}$  is **endogenous** to the system and depends on **deeply uncertain parameters** ( $b$  and  $q$ ), it should not be treated as a constant. Let's refine the formal model to reflect this dependency precisely.

## ♦ Refined Formal Model of the Lake Problem (with Endogenous $X_{\text{crit}}$ )

---

### 1. Overview

The lake problem is a **stochastic dynamic decision model** with **deep uncertainty** in ecological and economic parameters.

The decision-maker controls annual pollution emissions  $a_t$  to maximize long-term discounted benefits while keeping pollution below a dynamic eutrophication threshold  $X_{\text{crit}}(b, q)$ .

---

### 2. Time Horizon

$$t = 0, 1, 2, \dots, T$$


---

### 3. State Variable

Symbol	Definition	Domain
$X_t$	Lake pollution level at time $t$	$[0, X_{\max}]$

- **Initial condition:**  $X_0 = 0$
  - Represents accumulated phosphorus in the lake.
- 

### 4. Control (Decision) Variable

Symbol	Definition	Domain
$a_t$	Anthropogenic pollution emission (controlled by humans)	$[0, a_{\max}]$

---

### 5. Stochastic Variable

Symbol	Definition	Distribution
$\epsilon_t$	Natural (exogenous) pollution inflow	$\text{Lognormal}(\mu, \sigma^2)$

- Introduces **stochastic uncertainty** from random natural inflows.
- 

### 6. Transition Function

Pollution dynamics:

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

where:

- $\frac{X_t^q}{1+X_t^q}$ : natural recycling feedback,
  - $bX_t$ : natural removal of pollution.
- 

## 7. Objective Function

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

where:

- $\alpha$ : economic benefit per unit of pollution emission,
  - $\delta$ : discount rate,
  - Expectation  $\mathbb{E}[\cdot]$  is over stochastic inflows  $\epsilon_t$ .
- 

## 8. Eutrophication Threshold $X_{\text{crit}}$

### (a) Definition:

The lake becomes irreversibly eutrophic when its **natural recycling rate** equals its **natural removal rate**:

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

### (b) Solving for $X_{\text{crit}}$ :

$$X_{\text{crit}}^{q-1} = b(1 + X_{\text{crit}}^q)$$

or equivalently, the implicit function:

$$F(X_{\text{crit}}; b, q) = \frac{X_{\text{crit}}^q}{1 + X_{\text{crit}}^q} - bX_{\text{crit}} = 0$$

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

### (c) Implication:

- Because both  $b$  (removal rate) and  $q$  (recycling rate) are **deeply uncertain**,  $X_{\text{crit}}$  itself is **not fixed**.
  - Therefore, the eutrophication boundary **shifts** under different deep uncertainty realizations.
- 

## 9. Deep Uncertainties

Parameter	Description	Type of Uncertainty	Role
$\mu$	Mean of lognormal inflow	Deep	Affects stochastic inflow mean
$\sigma$	Std. dev. of inflow	Deep	Affects stochastic inflow variance
$b$	Natural removal rate	Deep	Affects resilience and $X_{\text{crit}}(b, q)$
$q$	Recycling rate	Deep	Affects feedback and $X_{\text{crit}}(b, q)$
$\delta$	Discount rate	Deep	Affects intertemporal valuation

Each of these parameters lies within an **uncertainty set**:

$$(\mu, \sigma, b, q, \delta) \in \mathcal{U}$$

where  $\mathcal{U}$  represents plausible parameter combinations (e.g., based on expert judgment or scenario analysis).

## 10. Constraint (Adaptive Safety Condition)

At each time  $t$ , the pollution level must remain below the **critical threshold determined by current uncertain parameters**:

$$X_t \leq X_{\text{crit}}(b, q), \quad \forall t$$

This defines a **robust constraint**, since  $X_{\text{crit}}$  varies across uncertainty realizations.

## 11. Model Summary Table

Component	Symbol	Description	Type
State	$X_t$	Lake pollution level	Endogenous
Decision	$a_t$	Pollution emission	Control
Stochastic input	$\epsilon_t \sim \text{Lognormal}(\mu, \sigma^2)$	Natural inflow	Random
Transition	$X_{t+1} = X_t + a_t + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t$	Pollution dynamics	Deterministic + stochastic
Objective	$\max \mathbb{E}[\sum \alpha a_t \delta^t]$	Discounted benefits	Economic
Threshold	$X_{\text{crit}}(b, q) : F(X_{\text{crit}}; b, q) = 0$	Eutrophication point	Endogenous
Constraint	$X_t \leq X_{\text{crit}}(b, q)$	Safety	Robust
Deep uncertainty	$\mu, \sigma, b, q, \delta$	Parameters with unknown probability	Structural uncertainty

## 12. Refined Problem Formulation

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

## 13. Interpretations

- **Stochastic uncertainty:** Random yearly fluctuations in natural inflows ( $\epsilon_t$ ).
- **Deep uncertainty:** Ambiguity in parameters ( $\mu, \sigma, b, q, \delta$ ) that define both system dynamics and the critical threshold.
- The problem can be approached via:
  - **Robust optimization** (optimize for worst-case parameter set),
  - **Info-gap decision theory**, or
  - **Exploratory modeling and scenario discovery** (test strategies across many plausible parameter sets).

Would you like me to take this next step and **express the model as a robust dynamic programming formulation**, showing how  $X_{\text{crit}}(b, q)$  modifies the Bellman recursion under deep uncertainty?



Various stakeholders approach this problem from different perspectives. Please identify four different perspectives related to this problem. These perspectives should share a common environment, along with global parameters and variables. For each perspective, also provide the corresponding model with the 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 exactly the right way to extend the *Lake Problem* into a **multi-perspective, multi-objective decision framework**, reflecting the diversity of stakeholder values and governance trade-offs.

Below is a **structured model of four stakeholder perspectives** that share the *same environment* and *global dynamics* (the lake system), but differ in **control focus, objectives, and risk attitudes**.

## Shared Environment and Global Structure

All stakeholders interact with the **same lake ecosystem** and thus share **global state variables, parameters, and uncertainties**.

# Global State and System Dynamics

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

**Shared components:**

Symbol	Description	Type
$X_t$	Lake pollution level at time $t$	State variable
$a_t$	Anthropogenic emissions (sum of all stakeholder emissions)	Aggregate control variable
$b$	Natural removal rate	Deeply uncertain parameter
$q$	Recycling rate	Deeply uncertain parameter
$\epsilon_t \sim \text{Lognormal}(\mu, \sigma^2)$	Natural inflow	Stochastic variable
$X_{\text{crit}}(b, q)$	Eutrophication threshold	Endogenous threshold
$\alpha$	Benefit-to-emission ratio	Shared economic coefficient
$\delta$	Discount rate	Shared temporal preference (uncertain)

The **pollution stock**  $X_t$  is a *shared environmental state* linking all perspectives.

## Perspective 1: Economic Growth Planner (Industry/Development Sector)

### Perspective

A local economic planner or industrial sector focuses on **maximizing economic benefits** from pollution-intensive activities, assuming environmental regulation is secondary or fixed.

### Control Variable

- $a_t^{(E)}$ : industrial pollution emissions.

### Transition Function

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

### Objective Function

$$\max_{a_t^{(E)}} f_{\text{economic}} = \mathbb{E} \left[ \sum_{t=0}^T \alpha_E a_t^{(E)} \delta^t \right]$$

### Constraint

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

## Interpretation

- Focus on **short- to medium-term profit maximization**.
- Assumes eutrophication is a constraint imposed externally (e.g., by regulation).
- Risk-neutral toward environmental thresholds.

## Perspective 2: Environmental Protection Agency (Ecological Regulator)

### Perspective

An environmental regulator or conservation authority aims to **preserve ecosystem health** and **avoid eutrophication**. Economic benefits are secondary.

### Control Variable

- $a_t^{(R)}$ : regulatory limit on total pollution (cap or tax control).

### Transition Function

Same shared system:

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

### Objective Function(s)

#### 1. Ecosystem integrity (primary):

$$\min_{a_t^{(R)}} f_{\text{ecological}} = \mathbb{E} \left[ \sum_{t=0}^T (X_t - X_{\text{crit}}(b, q))^2_+ \right]$$

(penalizes overshoot of critical pollution)

#### 2. Sustainability constraint:

$$\mathbb{P}(X_t < X_{\text{crit}}(b, q) \ \forall t) \geq 1 - \eta$$

(with confidence level  $1 - \eta$ )

## Interpretation

- Prioritizes environmental safety and resilience.
- Treats eutrophication as catastrophic and irreversible.
- Typically risk-averse and precautionary under deep uncertainty.

## Perspective 3: Social Welfare Planner (Integrated Policy Maker)

### Perspective

A central policymaker or community council seeks to **balance economic development and environmental protection**, valuing both in different dimensions rather than combining them in a single weighted sum.

### Control Variable

- $a_t^{(S)}$ : total social pollution policy (e.g., emission allowances, subsidies, or investment).

### Transition Function

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

### Objective Functions (Multi-objective)

#### 1. Economic welfare:

$$f_{\text{econ}} = \mathbb{E} \left[ \sum_{t=0}^T \alpha_S a_t^{(S)} \delta^t \right]$$

#### 2. Environmental safety:

$$f_{\text{env}} = -\mathbb{E} \left[ \sum_{t=0}^T (X_t - X_{\text{crit}}(b, q))^2_+ \right]$$

#### 3. Intergenerational equity:

$$f_{\text{equity}} = -\text{Var}_t(X_t)$$

(minimize variability in environmental quality over time)

### Interpretation

- Evaluates trade-offs across objectives rather than optimizing a single one.
- Could be implemented using **Pareto optimization** or **robust multi-objective decision analysis**.
- Aims for *sustainable compromise* between economy and ecology.

---

## Perspective 4: Local Community / Risk-Averse Stakeholders

### Perspective

Local residents, fishers, or farmers directly dependent on the lake's ecosystem prioritize **risk avoidance** and **long-term ecological stability** over average gains.

## Control Variable

- $a_t^{(C)}$ : local emissions or resource use intensity (could be smaller than industrial emission).

## Transition Function

Same as global system:

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

## Objective Functions

### 1. Minimize eutrophication risk (robust safety):

$$\min_{a_t^{(C)}} f_{\text{risk}} = \max_{(\mu, \sigma, b, q) \in \mathcal{U}} \mathbb{E} [(X_T - X_{\text{crit}}(b, q))_+]$$

(worst-case eutrophication penalty)

### 2. Maximize local livelihood stability:

$$\max_{a_t^{(C)}} f_{\text{livelihood}} = \mathbb{E} \left[ \sum_{t=0}^T \alpha_C a_t^{(C)} \delta^t \right] - \lambda \text{Var}(a_t^{(C)})$$

(stability in economic activity over time)

## Interpretation

- Emphasizes **robustness over optimality**.
- Prefers strategies that perform adequately across all plausible parameter realizations.
- Often aligns with adaptive management and precautionary principles.

## Comparative Summary of the Four Perspectives

Perspective	Control Variable	Objective(s)	Risk Attitude	Description
1. Economic Planner	$a_t^{(E)}$	Maximize economic benefit	Risk-neutral	Profit-focused growth
2. Environmental Regulator	$a_t^{(R)}$	Minimize ecological degradation	Risk-averse	Protect lake health
3. Social Planner	$a_t^{(S)}$	Multi-objective: economic + ecological + equity	Balanced	Integrative sustainable policy
4. Local Community	$a_t^{(C)}$	Minimize eutrophication risk; stabilize livelihoods	Robust, conservative	Long-term local resilience

## Common Features Across Perspectives

- **Shared state dynamics**  $X_{t+1} = X_t + a_t + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t$
- **Shared uncertainties** in  $(\mu, \sigma, b, q, \delta)$
- **Common environment** with feedback coupling between human activity and lake ecology
- Differ only by:
  - control locus (who decides and what variable they control),
  - objectives (single vs multi-dimensional),
  - treatment of uncertainty (expectation vs worst-case).

Would you like me to **formulate these four perspectives in a unified multi-agent dynamic framework** (e.g., as a *coupled dynamic game* or *multi-objective robust optimization problem*) showing possible conflicts or coordination mechanisms among the stakeholders?



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 — now we're moving from *parallel stakeholder models* to a **unified, internally consistent meta-model** that integrates all perspectives within a shared system.

This composite structure treats the *lake ecosystem* as a **common environment**, with each stakeholder exercising *independent but interacting controls*, and evaluating their own objectives modularly.

Below is the formal, fully consistent unified model.

## Unified Multi-Perspective Lake Problem Model

### 1. Global Environment (Shared by All Stakeholders)

#### State Variable

$$X_t \in [0, X_{\max}]$$

= pollution stock in the lake (e.g., phosphorus concentration).

#### Shared Dynamics

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

where

$$A_t = a_t^{(E)} + a_t^{(R)} + a_t^{(S)} + a_t^{(C)}$$

is the **aggregate anthropogenic emission** contributed by all perspectives.

## Stochastic and Deep Uncertainty

Symbol	Description	Type
$\epsilon_t \sim \text{Lognormal}(\mu, \sigma^2)$	Natural inflow (random)	Stochastic uncertainty
$\mu, \sigma, b, q, \delta$	Ecological & economic parameters	Deep uncertainty
$(\mu, \sigma, b, q, \delta) \in \mathcal{U}$	Feasible parameter set	Uncertainty set

## Endogenous Critical Threshold

The **eutrophication boundary**  $X_{\text{crit}}(b, q)$  is determined by the ecological feedback condition:

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

## Common Constraints

$$X_t \leq X_{\text{crit}}(b, q), \quad \forall t$$

(safety condition shared by all perspectives)

## 2. Stakeholder Perspectives (Modules)

Each perspective  $i \in \{\text{E, R, S, C}\}$  is modeled as a **modular agent** that:

- acts on its own **control variable**  $a_t^{(i)}$ ;
- evaluates **objectives**  $f^{(i)}(\cdot)$ ;
- treats all other controls  $a_t^{(-i)}$  as **exogenous inputs**;
- interacts via the **shared state variable**  $X_t$ .

Thus, for each perspective  $i$ :

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

where  $\bar{a}_t^{(-i)} = \sum_{j \neq i} a_t^{(j)}$ .

### 3. Perspective Modules

#### (1) Economic Growth Planner – Industry Sector (E)

**Control variable:**

$$a_t^{(E)} \in [0, a_{\max}^{(E)}]$$

**Transition function (for E's view):**

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

**Objective function:**

$$f^{(E)} = \mathbb{E} \left[ \sum_{t=0}^T \alpha_E a_t^{(E)} \delta^t \right]$$

**Constraint:**

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

#### (2) Environmental Regulator – Ecological Protection (R)

**Control variable:**

$$a_t^{(R)} \in [0, a_{\max}^{(R)}]$$

**Transition function:**

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

**Objective functions (multi-objective):**

##### 1. Ecological preservation

$$f_{\text{eco}}^{(R)} = \mathbb{E} \left[ \sum_{t=0}^T (X_t - X_{\text{crit}}(b, q))_+^2 \right]$$

(minimize deviation above critical threshold)

##### 2. Reliability requirement

$$\mathbb{P}(X_t < X_{\text{crit}}(b, q) \ \forall t) \geq 1 - \eta$$

#### (3) Social Welfare Planner – Balanced Policy Maker (S)

**Control variable:**

$$a_t^{(S)} \in [0, a_{\max}^{(S)}]$$

**Transition function:**

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

**Objective functions (three-dimensional):**

### 1. Economic welfare

$$f_{\text{econ}}^{(S)} = \mathbb{E} \left[ \sum_{t=0}^T \alpha_S a_t^{(S)} \delta^t \right]$$

### 2. Environmental safety

$$f_{\text{env}}^{(S)} = -\mathbb{E} \left[ \sum_{t=0}^T (X_t - X_{\text{crit}}(b, q))_+^2 \right]$$

### 3. Equity (stability)

$$f_{\text{equity}}^{(S)} = -\text{Var}_t(X_t)$$

These objectives can be analyzed through a **Pareto front** rather than combined.

---

## (4) Local Community – Risk-Averse Stakeholders (C)

**Control variable:**

$$a_t^{(C)} \in [0, a_{\max}^{(C)}]$$

**Transition function:**

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

**Objective functions (multi-objective, robust):**

### 1. Minimize worst-case eutrophication risk

$$f_{\text{risk}}^{(C)} = \max_{(\mu, \sigma, b, q) \in \mathcal{U}} \mathbb{E} [(X_T - X_{\text{crit}}(b, q))_+]$$

### 2. Maximize livelihood stability

$$f_{\text{livelhood}}^{(C)} = \mathbb{E} \left[ \sum_{t=0}^T \alpha_C a_t^{(C)} \delta^t \right] - \lambda_C \text{Var}(a_t^{(C)})$$


---

## 4. Unified Model Structure

The **composed model** combines all perspectives under shared dynamics and uncertainties:

$$\begin{aligned}
X_{t+1} &= X_t + \sum_{i \in \{E, R, S, C\}} a_t^{(i)} + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t, \\
\epsilon_t &\sim \text{Lognormal}(\mu, \sigma^2), \\
(\mu, \sigma, b, q, \delta) &\in \mathcal{U}, \\
X_t &\leq X_{\text{crit}}(b, q), \quad \forall t.
\end{aligned}$$

Each perspective  $i$  has its own control  $a_t^{(i)}$  and objective set  $\{f_k^{(i)}\}$ .

Other controls  $a_t^{(-i)}$  are treated as *exogenous* inputs to  $i$ 's decision-making.

## 5. Modular Evaluation Capability

This unified formulation allows:

- **Independent simulation or optimization** of each perspective  $i$ , holding other controls fixed.
- **Sequential or iterative evaluation** (e.g., policy coordination, Nash equilibrium analysis).
- **Scenario-based exploration** across deep uncertainty sets  $\mathcal{U}$ .
- **Multi-objective analysis** for perspectives with vector-valued objectives.

Mathematically:

$$\begin{aligned}
\text{For each } i : \quad & \max_{a_t^{(i)}} f^{(i)}(X_t, a_t^{(i)}, a_t^{(-i)}, \mu, \sigma, b, q, \delta) \\
\text{s.t.} \quad & X_{t+1} = X_t + a_t^{(i)} + \bar{a}_t^{(-i)} + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t.
\end{aligned}$$

## 6. Summary of Modular Components

Perspective	Control	Objectives	Treats other controls as...	Risk Attitude
E (Economic)	$a_t^{(E)}$	$f^{(E)} = \mathbb{E}[\sum \alpha_E a_t^{(E)} \delta^t]$	Exogenous	Neutral
R (Regulator)	$a_t^{(R)}$	$f_{\text{eco}}^{(R)}, f_{\text{reliability}}^{(R)}$	Exogenous	Risk-averse
S (Social)	$a_t^{(S)}$	$f_{\text{econ}}^{(S)}, f_{\text{env}}^{(S)}, f_{\text{equity}}^{(S)}$	Exogenous	Balanced
C (Community)	$a_t^{(C)}$	$f_{\text{risk}}^{(C)}, f_{\text{livelelihood}}^{(C)}$	Exogenous	Robust

## ✓ Final Unified Formulation (Compact Form)

Global dynamics:  $X_{t+1} = X_t + \sum_i a_t^{(i)} + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t,$   
 $\epsilon_t \sim \text{Lognormal}(\mu, \sigma^2), \quad (\mu, \sigma, b, q, \delta) \in \mathcal{U},$   
 $X_t \leq X_{\text{crit}}(b, q).$

For each perspective  $i$  :  $\max_{a_t^{(i)}} f_1^{(i)}, f_2^{(i)}, \dots$   
s.t.  $X_{t+1} = X_t + a_t^{(i)} + \bar{a}_t^{(-i)} + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t.$

## Interpretation

This unified framework:

- Ensures **state and uncertainty consistency** across all stakeholders;
- Encapsulates **heterogeneous objectives** within modular sub-models;
- Enables **independent or interactive evaluation** (e.g., cooperative, competitive, or policy coordination);
- Can serve as the foundation for:
  - **Robust multi-agent dynamic programming,**
  - **Scenario-based policy evaluation**, or
  - **Exploratory modeling under deep uncertainty** (e.g., MORDM or RDM frameworks).

Would you like me to **extend this unified model into a multi-agent dynamic game formulation** (e.g., with Nash equilibrium or cooperative Pareto-efficient coordination conditions)? That would allow explicit representation of how these perspectives strategically interact.



For the last three perspectives, are their control variables additive contributors to pollution? For example, the ecological regulator's control variable is the regulatory limit on total pollution, which should not be an additive contributor. Please refine your model specification.

Excellent — you're absolutely right to catch that subtle but crucial modeling inconsistency.

In the unified model, **not all control variables represent direct pollution emissions**:

- For the *industry* (economic planner),  $a_t^{(E)}$  **adds** to pollution.
- For the *regulator*,  $a_t^{(R)}$  **limits or mitigates** pollution.
- For the *social planner*,  $a_t^{(S)}$  **balances** policy interventions (e.g. incentives or investments).
- For the *community*,  $a_t^{(C)}$  reflects **local adaptive behaviors** that can either increase or decrease net pollution.

So, these controls must be differentiated by their **functional roles** in the system dynamics — not simply summed linearly.

Let's refine the model carefully and completely, preserving internal consistency and modularity.

---

## Refined Unified Multi-Perspective Lake Problem Model

### 1. Shared Global Environment

All perspectives share one physical system — the lake.

#### State Variable

$$X_t \in [0, X_{\max}]$$

= lake pollution (phosphorus concentration) at time  $t$ .

#### Stochastic & Deep Uncertainty

$$\epsilon_t \sim \text{Lognormal}(\mu, \sigma^2), \quad (\mu, \sigma, b, q, \delta) \in \mathcal{U}.$$

#### Endogenous Critical Threshold

$$\frac{X_{\text{crit}}^q}{1 + X_{\text{crit}}^q} = bX_{\text{crit}}, \quad X_{\text{crit}} = X_{\text{crit}}(b, q).$$

#### Shared Constraint

$$X_t \leq X_{\text{crit}}(b, q), \quad \forall t.$$


---

### 2. Perspective-Specific Controls and Their Roles

Each perspective  $i \in \{\text{E, R, S, C}\}$  acts on the shared environment through **its own control mechanism**  $u_t^{(i)}$ .

These controls enter the lake's pollution dynamics through a **net pollution inflow function**

$$a_t^{\text{net}} = \Phi(u_t^{(E)}, u_t^{(R)}, u_t^{(S)}, u_t^{(C)}),$$

which determines the **anthropogenic pollution contribution** to the lake.

We will now specify how each control affects  $a_t^{\text{net}}$ .

#### (1) Economic Growth Planner — Industry Sector (E)

- **Control:**  $u_t^{(E)} = a_t^{(E)} \geq 0$ : industrial emissions.
- **Role in environment:** *Direct additive source of pollution.*

$$\Phi_E = +a_t^{(E)}$$

## (2) Environmental Regulator — Ecological Protection (R)

- **Control:**  $u_t^{(R)} = r_t \in [0, 1]$ : regulatory stringency or emission cap fraction.
  - $r_t = 0$ : no regulation (maximum allowed emissions).
  - $r_t = 1$ : full restriction (no industrial pollution).
- **Role:** Reduces effective industrial emissions.

$$\Phi_R = -r_t a_t^{(E)}.$$

Hence, the regulator constrains pollution via a proportional reduction.

## (3) Social Welfare Planner — Policy & Investment (S)

- **Control:**  $u_t^{(S)} = s_t \geq 0$ : investment or mitigation policy intensity (e.g. pollution-control infrastructure, ecosystem restoration).
- **Role:** Removes or offsets part of the pollution stock or inflow.

$$\Phi_S = -\eta_S s_t,$$

where  $\eta_S \in (0, 1]$  is the efficiency of policy action in removing or neutralizing pollution.

## (4) Local Community — Adaptive Users (C)

- **Control:**  $u_t^{(C)} = c_t \in [-1, 1]$ : adaptive behavioral response — can reduce pollution (negative values) or increase it (positive).
- **Role:** Modifies net emissions through local practices.

$$\Phi_C = \gamma_C c_t,$$

where  $\gamma_C > 0$  scales local influence.

## 3. Refined Pollution Transition Function

Combining all controls, the net anthropogenic inflow is:

$$a_t^{\text{net}} = a_t^{(E)}(1 - r_t) + \gamma_C c_t - \eta_S s_t.$$

Then the lake pollution evolves as:

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

This function now correctly:

- adds emissions from the economy,
  - subtracts regulatory and mitigation actions,
  - includes stochastic natural inflow,
  - and remains internally consistent across perspectives.
- 

## 4. Perspective-Specific Modules

Each module  $i$  uses the *shared*  $X_t$  and the *shared dynamics*, but optimizes different objectives, treating others' controls as exogenous.

---

### (1) Economic Growth Planner (E)

$$\begin{aligned} \max_{a_t^{(E)} \geq 0} \quad f^{(E)} &= \mathbb{E} \left[ \sum_{t=0}^T \alpha_E a_t^{(E)} \delta^t \right], \\ \text{s.t.} \quad X_{t+1} &= X_t + a_t^{(E)}(1 - r_t) + \gamma_C c_t - \eta_S s_t + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t. \end{aligned}$$


---

### (2) Environmental Regulator (R)

$$\begin{aligned} \min_{r_t \in [0,1]} \quad f_{\text{eco}}^{(R)} &= \mathbb{E} \left[ \sum_{t=0}^T (X_t - X_{\text{crit}}(b, q))_+^2 \right], \\ \text{s.t.} \quad X_{t+1} &= X_t + a_t^{(E)}(1 - r_t) + \gamma_C c_t - \eta_S s_t + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t. \end{aligned}$$

Optionally also impose:

$$\mathbb{P}(X_t < X_{\text{crit}}(b, q) \ \forall t) \geq 1 - \eta.$$


---

### (3) Social Welfare Planner (S)

$$\begin{aligned} \max_{s_t \geq 0} \quad f_{\text{econ}}^{(S)} &= \mathbb{E} \left[ \sum_{t=0}^T \alpha_S (a_t^{(E)}(1 - r_t)) \delta^t \right], \\ f_{\text{env}}^{(S)} &= - \mathbb{E} \left[ \sum_{t=0}^T (X_t - X_{\text{crit}}(b, q))_+^2 \right], \\ f_{\text{equity}}^{(S)} &= - \text{Var}_t(X_t), \\ \text{s.t.} \quad X_{t+1} &= X_t + a_t^{(E)}(1 - r_t) + \gamma_C c_t - \eta_S s_t + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t. \end{aligned}$$


---

### (4) Local Community (C)

$$\begin{aligned}
\min_{c_t \in [-1,1]} \quad f_{\text{risk}}^{(C)} &= \max_{(\mu, \sigma, b, q) \in \mathcal{U}} \mathbb{E}[(X_T - X_{\text{crit}}(b, q))_+] , \\
\max_{c_t} \quad f_{\text{livelelihood}}^{(C)} &= \mathbb{E}\left[\sum_{t=0}^T \alpha_C(1 - |c_t|)\delta^t\right] - \lambda_C \text{Var}(c_t), \\
\text{s.t.} \quad X_{t+1} &= X_t + a_t^{(E)}(1 - r_t) + \gamma_C c_t - \eta_S s_t + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t.
\end{aligned}$$

## 5. Unified System Structure

$$\begin{aligned}
X_{t+1} &= X_t + [a_t^{(E)}(1 - r_t) + \gamma_C c_t - \eta_S s_t] + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t, \\
\epsilon_t &\sim \text{Lognormal}(\mu, \sigma^2), \\
(\mu, \sigma, b, q, \delta) &\in \mathcal{U}, \\
X_t &\leq X_{\text{crit}}(b, q), \quad X_0 = 0.
\end{aligned}$$

Each perspective  $i$  optimizes its own objective  $f^{(i)}$  over its own control  $u_t^{(i)}$ , treating others' controls as given.

## 6. Structural Summary

Perspective	Control	Role in Dynamics	Example Interpretation	Objective(s)	Effect on Pollution
E (Industry)	$a_t^{(E)}$	Adds emissions	Industrial output	Maximize profit	↑ increases
R (Regulator)	$r_t$	Multiplies $a_t^{(E)}$	Emission cap/tax	Minimize eutrophication	↓ reduces
S (Social planner)	$s_t$	Subtracts term	Investment in abatement or cleanup	Multi-objective (econ/env/equity)	↓ removes
C (Community)	$c_t$	Adds or subtracts	Local adaptive behavior	Min risk, max stability	± flexible

## 7. Key Properties of the Refined Model

- ✓ **Shared state and uncertainty structure** — ensures physical and ecological consistency.
- ✓ **Perspective modularity** — each decision-maker can be analyzed separately or jointly.
- ✓ **Control heterogeneity** — captures both additive and subtractive impacts on pollution.
- ✓ **Compatibility with dynamic game or coordinated governance frameworks.**

## Compact Representation

Global dynamics:  $X_{t+1} = X_t + F(u_t^{(E)}, u_t^{(R)}, u_t^{(S)}, u_t^{(C)}, X_t, \epsilon_t; b, q),$

$$F = a_t^{(E)}(1 - r_t) + \gamma_C c_t - \eta_S s_t + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t,$$
 $\epsilon_t \sim \text{Lognormal}(\mu, \sigma^2), (\mu, \sigma, b, q, \delta) \in \mathcal{U}.$

Each module  $i$  solves:

$$\max_{u_t^{(i)}} f^{(i)}(\text{own objectives}) \quad \text{s.t. } X_{t+1} = X_t + F(u_t^{(E)}, u_t^{(R)}, u_t^{(S)}, u_t^{(C)}, X_t, \epsilon_t).$$

Would you like me to **extend this unified, refined model** into a **multi-agent dynamic game** formulation — showing equilibrium or cooperative solution concepts (e.g., Nash, Stackelberg, or Pareto coordination)?



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.

Excellent — here's a **clean, modular Python implementation** of the *refined unified lake problem model* in object-oriented form.

This code defines a **common environment** shared by all perspectives, and four **stakeholder classes** (Economic, Regulator, Social, Community), each with its own control and objectives.

At the end, a top-level function `simulate_lake_model()` provides a **single interface**:

- It takes a dictionary of control trajectories and uncertain parameters,
- Simulates the pollution dynamics,
- Returns all stakeholders' objective values.

python

```
import numpy as np

# =====
# 1. Shared Lake Environment
# =====

class LakeEnvironment:
    """
    Represents the shared lake environment and its pollution dynamics.
    """

    def __init__(self, b, q, mu, sigma, delta, alpha_params, eta_S, gamma_C, X0=0.0, T=50):
        self.b = b                      # natural removal rate
        self.q = q                      # recycling rate
        self.mu = mu                     # lognormal inflow mean
        self.sigma = sigma                # lognormal inflow std
        self.delta = delta                # discount rate
        self.alpha = alpha_params         # dict of benefit coefficients (E, S, C)
```

```

self.eta_S = eta_S      # social mitigation efficiency
self.gamma_C = gamma_C  # community scaling factor
self.X0 = X0
self.T = T

def x_crit(self):
    """Compute the endogenous eutrophication threshold X_crit(b,q)."""
    # Solve for X such that  $X^q / (1 + X^q) = bX$ 
    # Use a simple numerical root finder
    from scipy.optimize import brentq
    f = lambda X: (X ** self.q) / (1 + X ** self.q) - self.b * X
    return brentq(f, 1e-6, 100.0)

def natural_inflow(self):
    """Draw stochastic natural inflow from a lognormal distribution."""
    return np.random.lognormal(mean=self.mu, sigma=self.sigma)

def step(self, X_t, controls):
    """
    One-step transition of the lake state.
    controls = dict with keys ['a_E', 'r', 's', 'c']
    """
    a_E = controls.get('a_E', 0.0)
    r = controls.get('r', 0.0)
    s = controls.get('s', 0.0)
    c = controls.get('c', 0.0)

    inflow_stochastic = self.natural_inflow()

    # Net anthropogenic pollution inflow
    a_net = a_E * (1 - r) + self.gamma_C * c - self.eta_S * s

    # Update pollution dynamics
    X_next = X_t + a_net + (X_t ** self.q) / (1 + X_t ** self.q) - self.b * X_t +
inflow_stochastic
    return X_next

# =====
# 2. Perspective Classes
# =====

class EconomicPlanner:
    """Industry / economic growth perspective."""
    def __init__(self, alpha_E):
        self.alpha_E = alpha_E

    def objective(self, controls, delta):
        a_E = np.array(controls['a_E'])
        T = len(a_E)
        return np.sum(self.alpha_E * a_E * (delta ** np.arange(T)))

class EnvironmentalRegulator:
    """Environmental protection perspective."""
    def ecological_loss(self, X, Xcrit):
        # Penalize eutrophication (excess pollution above threshold)
        return np.sum(np.maximum(0, X - Xcrit) ** 2)

    def reliability(self, X, Xcrit):
        # Probability (fraction of time steps) within safe bounds
        return np.mean(X < Xcrit)

class SocialPlanner:
    """Integrated welfare perspective (multi-objective)."""
    def __init__(self, alpha_S):
        self.alpha_S = alpha_S

    def economic_benefit(self, a_E, r, delta):
        # Economic benefit from effective emissions
        effective_emissions = np.array(a_E) * (1 - np.array(r))
        T = len(a_E)
        return np.sum(self.alpha_S * effective_emissions * (delta ** np.arange(T)))

```

```

def environmental_safety(self, X, Xcrit):
    return -np.sum(np.maximum(0, X - Xcrit) ** 2)

def equity(self, X):
    return -np.var(X)

class LocalCommunity:
    """Local adaptive perspective."""
    def __init__(self, alpha_C, lambda_C):
        self.alpha_C = alpha_C
        self.lambda_C = lambda_C

    def livelihood(self, c, delta):
        c = np.array(c)
        T = len(c)
        return np.sum(self.alpha_C * (1 - np.abs(c)) * (delta ** np.arange(T))) - self.lambda_C * np.var(c)

    def eutrophication_risk(self, X, Xcrit):
        return np.maximum(0, X[-1] - Xcrit) # risk proxy (final overshoot)

# =====
# 3. Unified Simulation Interface
# =====

def simulate_lake_model(control_dict, param_dict):
    """
    Simulate the unified lake problem under given controls and parameters.
    Inputs:
        control_dict: {
            'a_E': list[float],    # industrial emissions
            'r': list[float],      # regulation strength
            's': list[float],      # social mitigation effort
            'c': list[float]        # community adaptive response
        }
        param_dict: {
            'b': float, 'q': float, 'mu': float, 'sigma': float,
            'delta': float,
            'alpha_E': float, 'alpha_S': float, 'alpha_C': float,
            'eta_S': float, 'gamma_C': float, 'lambda_C': float,
            'T': int
        }
    Returns:
        results: dict with objective values for each perspective.
    """
    # Unpack parameters
    b = param_dict['b']
    q = param_dict['q']
    mu = param_dict['mu']
    sigma = param_dict['sigma']
    delta = param_dict['delta']
    alpha_E = param_dict['alpha_E']
    alpha_S = param_dict['alpha_S']
    alpha_C = param_dict['alpha_C']
    eta_S = param_dict['eta_S']
    gamma_C = param_dict['gamma_C']
    lambda_C = param_dict['lambda_C']
    T = param_dict['T']

    alpha_params = {'E': alpha_E, 'S': alpha_S, 'C': alpha_C}

    # Initialize environment and actors
    env = LakeEnvironment(b, q, mu, sigma, delta, alpha_params, eta_S, gamma_C, X0=0.0, T=T)
    econ = EconomicPlanner(alpha_E)
    regulator = EnvironmentalRegulator()
    social = SocialPlanner(alpha_S)
    community = LocalCommunity(alpha_C, lambda_C)

    # Compute critical threshold
    Xcrit = env.x_crit()

    # Simulate dynamics

```

```

X = np.zeros(T + 1)
for t in range(T):
    controls_t = {k: control_dict[k][t] for k in ['a_E', 'r', 's', 'c']}
    X[t + 1] = env.step(X[t], controls_t)

# =====
# Evaluate objectives
# =====

f_E = econ.objective(control_dict, delta)

f_R_eco = regulator.ecological_loss(X, Xcrit)
f_R_reliability = regulator.reliability(X, Xcrit)

f_S_econ = social.economic_benefit(control_dict['a_E'], control_dict['r'], delta)
f_S_env = social.environmental_safety(X, Xcrit)
f_S_equity = social.equity(X)

f_C_life = community.livelihood(control_dict['c'], delta)
f_C_risk = community.eutrophication_risk(X, Xcrit)

# Package results
results = {
    'EconomicPlanner': {'EconomicBenefit': f_E},
    'EnvironmentalRegulator': {'EcologicalLoss': f_R_eco,
                                'Reliability': f_R_reliability},
    'SocialPlanner': {'EconomicBenefit': f_S_econ,
                      'EnvironmentalSafety': f_S_env,
                      'Equity': f_S_equity},
    'LocalCommunity': {'Livelihood': f_C_life,
                       'EutrophicationRisk': f_C_risk},
    'FinalPollution': X[-1],
    'Trajectory': X
}

return results

```

## ✓ Key Features

Aspect	Description
<b>Shared environment</b>	<b>LakeEnvironment</b> handles common pollution dynamics and stochastic inflows.
<b>Perspective modularity</b>	Each class (EconomicPlanner, EnvironmentalRegulator, SocialPlanner, LocalCommunity) encapsulates its own objectives.
<b>Consistent transition function</b>	All controls interact through the unified pollution equation.
<b>Uncertainty integration</b>	Parameters like $b, q, \mu, \sigma, \delta$ passed via <b>param_dict</b> .
<b>Interface function</b>	<b>simulate_lake_model(control_dict, param_dict)</b> runs the simulation and returns all objectives.

## 📘 Example (illustrative, not executed)

```

python

controls = {
    'a_E': [0.5]*50,
    'r': [0.3]*50,
    's': [0.2]*50,
    'c': [0.0]*50
}

params = {

```

```
'b': 0.5, 'q': 2.0, 'mu': -2.0, 'sigma': 0.5,
'delta': 0.98,
'alpha_E': 1.0, 'alpha_S': 0.8, 'alpha_C': 0.6,
'eta_S^T': 0.3, 'gamma_C^T': 0.2, 'lambda_C^T': 0.1,
'T': 50
}

results = simulate_lake_model(controls, params)
```

This would return a dictionary with all objective values for each perspective (though we don't execute it here).

Would you like me to extend this model to **support Monte Carlo simulation over stochastic inflows** (averaging objectives over many random draws of  $\epsilon_t$ ) for robust evaluation under uncertainty?



There are two areas where your implementation could be improved: 1. the possible values of  $X_{\text{crit}}$  are in the range [0.01, 1.5]; 2. your implementation of the lognormal distribution is incorrect. Please refine your implementation.

Excellent observations — both are important for ensuring realism and correctness in simulation:

### 1. Range of $X_{\text{crit}}$ :

The eutrophication threshold  $X_{\text{crit}}$  should be restricted to the **ecologically meaningful range** [0.01, 1.5].

This avoids nonphysical solutions (very large or near-zero values) when solving  $\frac{X^q}{1+X^q} = bX$ .

### 2. Lognormal inflow generation:

In the original implementation, `np.random.lognormal(mean=μ, sigma=σ)` misinterpreted  $\mu$  and  $\sigma$  as the *mean and standard deviation of the lognormal variable*, but NumPy expects them as the *mean and standard deviation of the underlying normal distribution*.

To correctly sample a lognormal distribution with **desired mean and variance** of the actual inflow ( $\mathbb{E}[\epsilon_t] = \mu$ ,  $\text{Var}[\epsilon_t] = \sigma^2$ ), we need to transform them to the parameters of the underlying normal distribution.

## Refined Unified Modular Implementation

python

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

# =====
# 1. Shared Lake Environment
# =====

class LakeEnvironment:
    """
    Represents the shared lake environment and its pollution dynamics.
    Handles stochastic inflow and endogenous eutrophication threshold.
    """

    def __init__(self, ...):
```

```

def __init__(self, b, q, mu, sigma, delta, alpha_params, eta_S, gamma_C, X0=0.0, T=50):
    self.b = b                      # natural removal rate
    self.q = q                        # recycling rate
    self.mu = mu                      # mean of actual lognormal inflow
    self.sigma = sigma                # std dev of actual lognormal inflow
    self.delta = delta                # discount rate
    self.alpha = alpha_params          # dict of benefit coefficients
    self.eta_S = eta_S                # social mitigation efficiency
    self.gamma_C = gamma_C            # community scaling factor
    self.X0 = X0
    self.T = T

# -----
# Compute endogenous eutrophication threshold
# -----
def x_crit(self):
    """
    Compute X_crit(b,q) within [0.01, 1.5], where X^q/(1+X^q) = bX.
    """
    f = lambda X: (X ** self.q) / (1 + X ** self.q) - self.b * X
    try:
        return brentq(f, 0.01, 1.5)
    except ValueError:
        # If no root in range, clamp to nearest boundary
        return 0.01 if f(0.01) * f(1.5) > 0 else 1.5

# -----
# Generate stochastic natural inflow
# -----
def natural_inflow(self):
    """
    Draw stochastic inflow ε_t from a lognormal distribution
    with mean=self.mu and std=self.sigma (in real space).
    """
    # Convert desired mean (μ) and std (σ) to log-space parameters
    # Let m, s be mean and std of underlying normal distribution:
    # μ = exp(m + s^2 / 2), σ^2 = (exp(s^2) - 1) * exp(2m + s^2)
    variance = self.sigma ** 2
    s2 = np.log(1 + variance / (self.mu ** 2))
    s = np.sqrt(s2)
    m = np.log(self.mu) - 0.5 * s2
    return np.random.lognormal(mean=m, sigma=s)

# -----
# One-step state transition
# -----
def step(self, X_t, controls):
    """
    One-step pollution update:
    controls = dict with keys ['a_E', 'r', 's', 'c']
    """
    a_E = controls.get('a_E', 0.0)
    r = controls.get('r', 0.0)
    s = controls.get('s', 0.0)
    c = controls.get('c', 0.0)

    inflow_stochastic = self.natural_inflow()

    # Net anthropogenic pollution inflow
    a_net = a_E * (1 - r) + self.gamma_C * c - self.eta_S * s

    # Pollution dynamics
    X_next = X_t + a_net + (X_t ** self.q) / (1 + X_t ** self.q) - self.b * X_t +
    inflow_stochastic
    return max(X_next, 0.0)  # ensure nonnegative pollution

# =====
# 2. Perspective Classes
# =====

class EconomicPlanner:
    """Industry / economic growth perspective."""
    def __init__(self, alpha_E):
        self.alpha_E = alpha_E

```

```

def objective(self, controls, delta):
    a_E = np.array(controls['a_E'])
    T = len(a_E)
    return np.sum(self.alpha_E * a_E * (delta ** np.arange(T)))

class EnvironmentalRegulator:
    """Environmental protection perspective."""
    def ecological_loss(self, X, Xcrit):
        return np.sum(np.maximum(0, X - Xcrit) ** 2)

    def reliability(self, X, Xcrit):
        return np.mean(X < Xcrit)

class SocialPlanner:
    """Integrated welfare perspective (multi-objective)."""
    def __init__(self, alpha_S):
        self.alpha_S = alpha_S

    def economic_benefit(self, a_E, r, delta):
        eff = np.array(a_E) * (1 - np.array(r))
        T = len(a_E)
        return np.sum(self.alpha_S * eff * (delta ** np.arange(T)))

    def environmental_safety(self, X, Xcrit):
        return -np.sum(np.maximum(0, X - Xcrit) ** 2)

    def equity(self, X):
        return -np.var(X)

class LocalCommunity:
    """Local adaptive perspective."""
    def __init__(self, alpha_C, lambda_C):
        self.alpha_C = alpha_C
        self.lambda_C = lambda_C

    def livelihood(self, c, delta):
        c = np.array(c)
        T = len(c)
        return np.sum(self.alpha_C * (1 - np.abs(c)) * (delta ** np.arange(T))) - self.lambda_C * np.var(c)

    def eutrophication_risk(self, X, Xcrit):
        return np.maximum(0, X[-1] - Xcrit)

# =====
# 3. Unified Simulation Interface
# =====

def simulate_lake_model(control_dict, param_dict):
    """
    Simulate the unified lake model with given controls and uncertain parameters.

    Inputs:
        control_dict: dict of control time series
            {'a_E': [...], 'r': [...], 's': [...], 'c': [...]}
        param_dict: dict of uncertain parameters and constants
            {'b', 'q', 'mu', 'sigma', 'delta',
             'alpha_E', 'alpha_S', 'alpha_C',
             'eta_S', 'gamma_C', 'lambda_C', 'T'}
    """

    Returns:
        results: dict of all objective values and final pollution trajectory.
    """

    # Unpack parameters
    b = param_dict['b']
    q = param_dict['q']
    mu = param_dict['mu']
    sigma = param_dict['sigma']
    delta = param_dict['delta']

```

```

alpha_E = param_dict['alpha_E']
alpha_S = param_dict['alpha_S']
alpha_C = param_dict['alpha_C']
eta_S = param_dict['eta_S']
gamma_C = param_dict['gamma_C']
lambda_C = param_dict['lambda_C']
T = param_dict['T']

alpha_params = {'E': alpha_E, 'S': alpha_S, 'C': alpha_C}

# Initialize environment and perspectives
env = LakeEnvironment(b, q, mu, sigma, delta, alpha_params, eta_S, gamma_C, X0=0.0, T=T)
econ = EconomicPlanner(alpha_E)
regulator = EnvironmentalRegulator()
social = SocialPlanner(alpha_S)
community = LocalCommunity(alpha_C, lambda_C)

# Compute critical threshold (bounded)
Xcrit = env.x_crit()

# Simulate pollution trajectory
X = np.zeros(T + 1)
for t in range(T):
    controls_t = {k: control_dict[k][t] for k in ['a_E', 'r', 's', 'c']}
    X[t + 1] = env.step(X[t], controls_t)

# Compute objective values
f_E = econ.objective(control_dict, delta)
f_R_eco = regulator.ecological_loss(X, Xcrit)
f_R_rel = regulator.reliability(X, Xcrit)
f_S_econ = social.economic_benefit(control_dict['a_E'], control_dict['r'], delta)
f_S_env = social.environmental_safety(X, Xcrit)
f_S_equity = social.equity(X)
f_C_life = community.livelihood(control_dict['c'], delta)
f_C_risk = community.eutrophication_risk(X, Xcrit)

# Return results
results = {
    'EconomicPlanner': {'EconomicBenefit': f_E},
    'EnvironmentalRegulator': {'EcologicalLoss': f_R_eco,
                                'Reliability': f_R_rel},
    'SocialPlanner': {'EconomicBenefit': f_S_econ,
                      'EnvironmentalSafety': f_S_env,
                      'Equity': f_S_equity},
    'LocalCommunity': {'Livelihood': f_C_life,
                       'EutrophicationRisk': f_C_risk},
    'FinalPollution': X[-1],
    'Trajectory': X
}

return results

```

## Improvements Explained

Issue	Correction
1. Valid range for $X_{\text{crit}}$	Added <code>brentq</code> solver restricted to [0.01, 1.5]; clamps to boundary if no root found.
2. Lognormal sampling	Now correctly converts user-supplied <i>mean</i> and <i>std</i> in real space to underlying normal parameters before calling <code>np.random.lognormal(mean=m, sigma=s)</code> .
3. Stability	Added <code>max(X_next, 0.0)</code> to prevent negative pollution levels.
4. Readability & modularity	Classes and methods remain modular and easily expandable for multi-run or policy experimentation.

Would you like me to extend this version so it can **average results over multiple stochastic realizations** (Monte Carlo runs) to estimate expected objective values robustly under uncertainty?