



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 model** of the *Lake Problem*, structured in the style of a stochastic dynamic optimization model that captures both **stochastic** and **deep uncertainties**.

## 1. Model Overview

The lake problem can be formulated as a **stochastic dynamic decision model** where the decision-maker seeks to maximize discounted economic benefits from pollution while avoiding irreversible eutrophication. The dynamics are governed by the pollution accumulation in the lake.

## 2. State Variables

Symbol	Description
$X_t$	Lake pollution stock at time $t$ . It represents the level of accumulated pollutants in the lake.

Symbol	Description
$X_0 = 0$	Initial condition: the lake starts unpolluted.

## 3. Decision (Control) Variable

Symbol	Description
$a_t$	Anthropogenic (human-caused) pollution emission rate at time $t$ . Decision-makers choose this level each year.

## 4. Stochastic Variable

Symbol	Description
$\epsilon_t$	Natural inflow of pollution at time $t$ . Random variable following a log-normal distribution: $\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$ .

## 5. System Transition (State Evolution)

The pollution dynamics are given by:

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

where:

- $\frac{X_t^q}{1 + X_t^q}$  models the **nonlinear recycling effect** (the lake's natural feedback).
- $bX_t$  captures **natural removal** (e.g., sedimentation, outflow, decay).

## 6. Objective Function (Economic Benefit)

The total discounted economic benefit is:

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

where:

- $\alpha$ : benefit per unit of pollution emission (benefit-to-pollution ratio),
- $\delta$ : discount factor (intertemporal preference),
- $T$ : time horizon.

## 7. Constraint (Eutrophication Threshold)

To avoid irreversible eutrophication, the pollution level must remain below a critical threshold:

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

where  $X_{\text{crit}}$  is defined implicitly by the equality:

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

## 8. Parameters (Nominal Values / Constants)

Symbol	Description
$b$	Natural removal rate of the lake.
$q$	Natural recycling rate of the lake.
$\alpha$	Benefit-to-pollution ratio.
$\delta$	Discount rate.
$X_{\text{crit}}$	Eutrophication threshold.
$T$	Planning horizon (number of decision periods).

## 9. Uncertainties

### (a) Stochastic Uncertainty

- Source: random natural pollution inflows.
- Modeled via the random variable  $\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$ .

### (b) Deep Uncertainty

Deep uncertainty arises when even the *distributions* or *parameter values* are unknown or disputed:

- Mean of natural inflow:  $\mu$
- Variance (or standard deviation):  $\sigma$
- Natural removal rate:  $b$
- Natural recycling rate:  $q$
- Discount rate:  $\delta$

These parameters may be represented as **uncertain sets** or **scenarios**, e.g.:

$$\theta = \{\mu, \sigma, b, q, \delta\} \in \Theta$$

where  $\Theta$  is a set of plausible parameter combinations.

## 10. Formal Model Summary

## Objective:

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

## Subject to:

$$\begin{cases} X_{t+1} = X_t + a_t + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t, & \forall t \in T, \\ X_t < X_{\text{crit}}, & \forall t \in T, \\ a_t \geq 0, \quad X_0 = 0, \\ \epsilon_t \sim \text{LogNormal}(\mu, \sigma^2). \end{cases}$$

## 11. Interpretation and Use

This model can be used for:

- **Stochastic Dynamic Programming (SDP)** — to compute optimal emission policies  $a_t^*(X_t)$ .
- **Robust Decision Making (RDM)** — to test strategies under deep uncertainty across many plausible parameter sets  $\Theta$ .
- **Adaptive Management** — updating policies as new observations of  $X_t$  and  $\epsilon_t$  become available.

Would you like me to extend this into a **Markov Decision Process (MDP) formulation** (with state space, actions, transitions, and reward function) suitable for computational solution (e.g., dynamic programming or reinforcement learning)?



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 — you're asking for a structured, **multi-perspective formalization** of the *Lake Problem*, where different stakeholders share the same physical environment (the lake and its dynamics) but differ in their **objectives, controls, and decision priorities**.

Below, I'll identify **four stakeholder perspectives**, then formalize each as a distinct model, with clear specification of its **control variables, state and transition dynamics, objective functions, and constraints**.

## Shared Common Environment

All perspectives share the same **physical and environmental model** — the *lake dynamics, state variable, and uncertainties*.

## Global State Variable

$X_t$  : Lake pollution stock at time  $t$

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

where:

- $a_t$ : anthropogenic emissions (control variable in most models),
- $b$ : natural removal rate,
- $q$ : recycling rate,
- $\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$ : natural inflow (stochastic).

## Global Constraints

$$X_t \geq 0, \quad X_t < X_{\text{crit}} \quad \forall t$$

## Global Parameters

$$\alpha, b, q, \delta, \mu, \sigma, X_{\text{crit}}$$

## 1 Economic Planner's Perspective (Growth-Oriented)

### Motivation

The local government or economic authority seeks to **maximize total discounted economic benefit** from industrial and agricultural activity, while respecting environmental limits.

### Control Variable

$a_t$  : anthropogenic pollution emissions (economic activity level)

### State Transition

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

### Objective Function

$$\max_{a_t} f_{\text{economic}} = \sum_{t=0}^T \alpha a_t \delta^t$$

### Constraints

$$X_t < X_{\text{crit}}, \quad a_t \geq 0$$

## Key Perspective

- Sees pollution as a byproduct of necessary growth.
- May accept moderate pollution risks for short-term gain.

## 2 Environmental Regulator's Perspective (Risk-Averse)

### Motivation

The environmental agency aims to **minimize the risk of eutrophication** and **preserve long-term ecological health**, potentially at the cost of lower economic benefit.

### Control Variable

$a_t$  : allowable anthropogenic pollution emission (regulated)

### State Transition

(same as above)

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

### Objective Function

Minimize expected pollution and eutrophication probability:

#### 1. Expected Pollution Minimization

$$\min_{a_t} f_{\text{pollution}} = \mathbb{E} \left[ \sum_{t=0}^T X_t \right]$$

#### 2. Eutrophication Risk Minimization

$$\min_{a_t} f_{\text{risk}} = \Pr(X_t \geq X_{\text{crit}}) \quad \forall t$$

## Key Perspective

- Prioritizes environmental safety.
- May impose caps, permits, or taxes on emissions.
- Sees uncertainty (especially deep uncertainty) as a major concern.

## 3 Community / Public Health Perspective (Social Welfare)

### Motivation

The local community or public health authority seeks to **balance economic welfare and ecosystem health**, emphasizing social sustainability and intergenerational equity.

## Control Variables

- $a_t$  : community-supported pollution (economic participation)
- $u_t$  : pollution mitigation effort (e.g., clean tech investment)

## State Transition

Pollution can be offset by mitigation:

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

## Objective Functions

### 1. Economic Well-being

$$f_1 = \sum_{t=0}^T \alpha a_t \delta^t$$

### 2. Health and Environmental Quality

$$f_2 = - \sum_{t=0}^T X_t \delta^t$$

### 3. Equity / Fairness

$$f_3 = -\text{Var}(X_t) \quad (\text{reduce volatility and intergenerational inequality})$$

## Key Perspective

- Multi-objective optimization between economic benefit, health, and fairness.
- Often supports adaptive or participatory governance policies.

## 4 Global or Scientific Perspective (Sustainability / Robustness)

### Motivation

Researchers or global policymakers aim to identify **robust strategies** that perform well under deep uncertainty — focusing on **system resilience** and **sustainability**.

### Control Variable

- $a_t$  : policy-level emission control or adaptive rule

### Uncertain Parameters

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

treated as **deeply uncertain**, not known precisely.

## State Transition (Uncertain Parameters)

$$X_{t+1} = X_t + a_t + \frac{X_t^{q(\theta)}}{1 + X_t^{q(\theta)}} - b(\theta)X_t + \epsilon_t(\mu, \sigma)$$

## Objectives

Multi-objective formulation across **performance, robustness, and sustainability**:

### 1. Expected Performance

$$f_1 = \mathbb{E}_\epsilon \left[ \sum_{t=0}^T \alpha a_t \delta^t \right]$$

### 2. Robustness (minimize worst-case loss across uncertainty set $\Theta$ )

$$f_2 = \min_{\theta \in \Theta} \left( \sum_{t=0}^T \alpha a_t \delta^t \right)$$

### 3. Sustainability (avoid threshold crossing under all plausible futures)

$$f_3 = - \max_{\theta \in \Theta} \Pr(X_t \geq X_{\text{crit}})$$

## Key Perspective

- Evaluates strategies under **many plausible futures** (robust decision-making or info-gap frameworks).
- Focuses on **trade-offs between performance and resilience**.

## Summary Table

Perspective	Decision Variable(s)	Objective(s)	Key Focus
1. Economic Planner	$a_t$	Maximize total economic benefit	Growth and revenue
2. Environmental Regulator	$a_t$	Minimize pollution and eutrophication risk	Environmental protection
3. Community / Public Health	$a_t, u_t$	Maximize economic benefit; minimize pollution; ensure fairness	Social welfare, intergenerational justice
4. Global / Scientific (Robust)	$a_t$	Maximize expected benefit, minimize worst-case loss, ensure sustainability	Resilience and robustness under deep uncertainty

Would you like me to extend this into a **multi-agent formulation** (e.g., game-theoretic or multi-objective coordination model) to show how these four perspectives could interact or conflict within the

same system?



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.

Excellent — this is a sophisticated modeling step.

We'll now **compose a unified lake system model** that integrates all four stakeholder perspectives into a single **multi-perspective dynamic system**, maintaining internal consistency and modularity.

The unified model will:

1. Share **one common environment** (the lake and its pollution dynamics).
2. Represent each stakeholder's **control variables** and **objectives** explicitly.
3. Treat **other stakeholders' controls as exogenous** when evaluating one perspective.
4. Maintain **clear modularity**, so that each perspective's policy can be simulated or optimized independently while remaining coupled through shared system states.

## Unified Multi-Perspective Lake System Model

### 1. Common Environment and Shared States

All perspectives interact through a **shared environmental state**  $X_t$ , representing lake pollution.

#### State Variable

$X_t$  : Pollution stock in the lake at time  $t$

#### Environmental Transition Function

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

where:

- $A_t$  = **aggregate anthropogenic net emissions** at time  $t$ ,
- $\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$ : stochastic natural inflow,
- $b$ : natural removal rate,
- $q$ : recycling rate.

## Aggregate Emissions (Endogenous Composition)

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

where:

- $a_t^{(E)}$ : emissions from **economic planner**,
- $a_t^{(R)}$ : emissions allowed by **environmental regulator** (e.g., baseline industrial quota),
- $u_t^{(C)}$ : mitigation effort by **community** (reduces effective emissions),
- $a_t^{(S)}$ : global or scientific policy intervention (e.g., subsidies or global offsets).

Each perspective controls its variable independently but interacts through  $X_t$ .

## 2. Common Parameters and Uncertainties

Symbol	Description	Type
$b$	Natural removal rate	Deep uncertainty
$q$	Natural recycling rate	Deep uncertainty
$\mu, \sigma$	Parameters of lognormal natural inflow	Deep uncertainty
$\delta$	Discount rate	Deep uncertainty
$\alpha$	Benefit-to-pollution ratio	Constant
$X_{\text{crit}}$	Eutrophication threshold	Constant
$\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$	Natural stochastic inflow	Stochastic uncertainty

## 3. Shared Constraints

$$X_t \geq 0, \quad X_t < X_{\text{crit}}, \quad a_t^{(E)}, a_t^{(R)}, u_t^{(C)}, a_t^{(S)} \geq 0$$

## 4 Modular Perspective Submodels

Each perspective observes the same state  $X_t$  and global parameters, but controls only its own variables and optimizes its own objectives.

### Perspective 1: Economic Planner (E)

#### Control Variable

$a_t^{(E)}$  : anthropogenic emissions from economic activities

#### Objective

$$\max_{a_t^{(E)}} f_{\text{economic}}^{(E)} = \sum_{t=0}^T \alpha a_t^{(E)} \delta^t$$

## Transition Function (shared)

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

## Perspective Assumptions

- Takes  $a_t^{(R)}$ ,  $u_t^{(C)}$ ,  $a_t^{(S)}$  as **exogenous**.
  - Focuses purely on maximizing discounted profit.
- 

## Perspective 2: Environmental Regulator (R)

### Control Variable

$a_t^{(R)}$  : allowable emissions cap or regulatory adjustment

### Objectives

#### 1. Pollution Minimization

$$\min_{a_t^{(R)}} f_{\text{pollution}}^{(R)} = \mathbb{E} \left[ \sum_{t=0}^T X_t \right]$$

#### 2. Eutrophication Risk Minimization

$$\min_{a_t^{(R)}} f_{\text{risk}}^{(R)} = \Pr(X_t \geq X_{\text{crit}}) \quad \forall t$$

## Perspective Assumptions

- Takes  $a_t^{(E)}$ ,  $u_t^{(C)}$ ,  $a_t^{(S)}$  as **exogenous**.
  - Regulates emissions to ensure environmental safety.
- 

## Perspective 3: Community / Public Health (C)

### Control Variable

$u_t^{(C)}$  : mitigation or cleanup effort

### Objectives

#### 1. Economic Wellbeing

$$f_1^{(C)} = \sum_{t=0}^T \alpha a_t^{(E)} \delta^t$$

## 2. Environmental Health

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

## 3. Equity / Stability

$$f_3^{(C)} = -\text{Var}(X_t)$$

### Perspective Assumptions

- Takes  $a_t^{(E)}, a_t^{(R)}, a_t^{(S)}$  as exogenous.
  - Chooses  $u_t^{(C)}$  to balance wellbeing, health, and fairness.
- 

## Perspective 4: Global / Scientific (S)

### Control Variable

$a_t^{(S)}$  : global emission adjustment or subsidy policy

### Uncertain Parameters

$$\theta = \{\mu, \sigma, b, q, \delta\} \in \Theta$$

### Objectives

#### 1. Expected Performance

$$f_1^{(S)} = \mathbb{E}_\epsilon \left[ \sum_{t=0}^T \alpha A_t \delta^t \right]$$

#### 2. Robustness (Worst-Case Performance)

$$f_2^{(S)} = \min_{\theta \in \Theta} \left[ \sum_{t=0}^T \alpha A_t \delta^t \right]$$

#### 3. Sustainability (Resilience Constraint)

$$f_3^{(S)} = - \max_{\theta \in \Theta} \Pr(X_t \geq X_{\text{crit}})$$

### Perspective Assumptions

- Treats all other controls as exogenous.
- Focuses on robustness and sustainability under deep uncertainty.



## Unified Mathematical Representation

$$\begin{cases} X_{t+1} = X_t + (a_t^{(E)} + a_t^{(R)} - u_t^{(C)} + a_t^{(S)}) + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t, \\ \epsilon_t \sim \text{LogNormal}(\mu, \sigma^2), \\ a_t^{(E)}, a_t^{(R)}, u_t^{(C)}, a_t^{(S)} \geq 0, \quad X_t < X_{\text{crit}}. \end{cases}$$

Each perspective  $i \in \{E, R, C, S\}$  optimizes its own objective(s):

$$\max_{c_t^{(i)}} f^{(i)}(\{c_t^{(i)}\}, X_t, \Theta)$$

where  $c_t^{(i)}$  is the vector of that perspective's control variables.

## Gears Modular Evaluation Mechanism

Component	Description
<b>Shared Environment Module</b>	Simulates lake pollution $X_t$ based on combined controls and stochastic inflows.
<b>Perspective Modules</b>	Each stakeholder has its own optimizer for its control variable(s).
<b>Coupling Mechanism</b>	All optimizers feed into $A_t$ , influencing future $X_t$ .
<b>Evaluation Layer</b>	Allows independent or joint optimization, scenario simulation, or sensitivity analysis.



## Interpretation

This unified model supports:

- Single-perspective evaluation:** hold others' controls constant, optimize one.
- Co-evolutionary analysis:** simulate joint decision dynamics.
- Policy coordination studies:** examine trade-offs or cooperative outcomes.
- Robust and multi-objective exploration:** evaluate performance across deep uncertainty scenarios.

## ✓ Summary: Unified Model Schema

Layer	Symbol / Equation	Shared or Modular
<b>State</b>	$X_t$	Shared
<b>Transition</b>	$X_{t+1} = X_t + A_t + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t$	Shared
<b>Uncertainty</b>	$\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2), \Theta = \{\mu, \sigma, b, q, \delta\}$	Shared
<b>Controls</b>	$a_t^{(E)}, a_t^{(R)}, u_t^{(C)}, a_t^{(S)}$	Modular

Layer	Symbol / Equation	Shared or Modular
Objectives	$f^{(E)}, f^{(R)}, f^{(C)}, f^{(S)}$	Modular
Constraints	$X_t < X_{\text{crit}}$ , all controls $\geq 0$	Shared
Coupling Variable	$A_t = a_t^{(E)} - u_t^{(C)} + a_t^{(R)} + a_t^{(S)}$	Shared
Evaluation	Individual or joint optimization	Modular

Would you like me to take this unified model one step further — e.g., **formulate it as a multi-agent dynamic game (generalized Nash equilibrium)** or as a **multi-objective optimization framework (for cooperative planning)**? Both are natural next steps depending on whether you want to study *conflict* or *coordination*.



There are two areas where your model specification could be improved: 1. if the environmental regulator's control variable is the allowable emissions cap, it should not be added to the other control variables; 2.  $X_{\text{crit}}$  is not a constant because it depends on other deeply uncertain parameters. Please refine your model specification.

Excellent — you are absolutely right on both points.

Let's correct and refine the unified model carefully and systematically, ensuring **mathematical consistency, semantic clarity, and modularity** across all perspectives.

## Refined Unified Multi-Perspective Lake System Model

This version corrects:

### 1. Environmental Regulator's control variable —

The regulator sets a *cap* (an upper limit) on allowable emissions rather than directly adding to emissions.

→ Hence,  $a_t^{(E)} \leq a_t^{(R)}$ , and the regulator's control influences the *constraint*, not the additive emission term.

### 2. Dynamic eutrophication threshold —

$X_{\text{crit}}$  depends on the natural parameters  $b$  and  $q$ , which are **deeply uncertain**, and therefore must be modeled as an **endogenous function**:

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

defined implicitly by the condition where **natural recycling equals natural removal**:

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

## 1 Common Environment

All perspectives share the same physical lake environment and pollution dynamics.

## State Variable

$X_t$  : Lake pollution stock at time  $t$

## Stochastic Natural Pollution

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

with mean  $\mu$  and variance  $\sigma^2$  — both deeply uncertain.

## 2 Pollution Transition Function

The **shared dynamic system** evolves as:

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

where

$$a_t^{(\text{eff})} = a_t^{(E)} - u_t^{(C)} + a_t^{(S)}$$

is the **effective anthropogenic pollution** entering the lake after community mitigation and global adjustments.

### Constraint Imposed by Regulator:

$$a_t^{(E)} \leq a_t^{(R)}.$$

Thus, the regulator does not contribute emissions directly but sets the *policy constraint* bounding economic emissions.

## 3 Shared and Uncertain Parameters

Symbol	Description	Type
$b$	Natural removal rate	Deep uncertainty
$q$	Natural recycling rate	Deep uncertainty
$\mu, \sigma$	Natural inflow distribution parameters	Deep uncertainty
$\delta$	Discount rate	Deep uncertainty
$\alpha$	Benefit-to-pollution ratio	Constant
$X_{\text{crit}}(b, q)$	Critical eutrophication threshold	Endogenous (function of $b, q$ )

### Eutrophication Boundary Condition

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

This makes the eutrophication risk itself a **function of deep uncertainty**.

---

## 4 Shared Constraints

$$\begin{cases} X_t \geq 0, \\ a_t^{(E)}, u_t^{(C)}, a_t^{(S)}, a_t^{(R)} \geq 0, \\ a_t^{(E)} \leq a_t^{(R)}, \\ X_t < X_{\text{crit}}(b, q). \end{cases}$$


---

## 5 Perspective-Specific Submodels

Each stakeholder interacts through the same environment and shared state but controls only its own decision variables, treating others' actions as **exogenous inputs**.

---

### (1) Economic Planner – Growth Perspective

#### Control Variable

$a_t^{(E)}$  : Anthropogenic emissions from economic activity

#### Objective

$$\max_{a_t^{(E)}} \quad f^{(E)} = \sum_{t=0}^T \alpha a_t^{(E)} \delta^t$$

#### Subject to

$a_t^{(E)} \leq a_t^{(R)}$ , and shared system dynamics and constraints.

#### Interpretation

The planner maximizes economic output while obeying regulatory and environmental constraints.

---

### (2) Environmental Regulator – Risk-Averse Perspective

#### Control Variable

$a_t^{(R)}$  : Emission cap or regulatory limit.

#### Objectives

1. Minimize expected pollution load

$$f_1^{(R)} = \mathbb{E} \left[ \sum_{t=0}^T X_t \right]$$

## 2. Minimize probability of eutrophication

$$f_2^{(R)} = \Pr(X_t \geq X_{\text{crit}}(b, q))$$

### Subject to

Shared system dynamics; recognizes that  $a_t^{(E)} \leq a_t^{(R)}$ .

### Interpretation

The regulator determines allowable emissions such that environmental safety and sustainability are preserved under uncertainty.

---

## (3) Community / Public Health – Social Welfare Perspective

### Control Variable

$u_t^{(C)}$  : Mitigation or cleanup effort (reduces net emissions).

### Objectives

#### 1. Economic Well-Being

$$f_1^{(C)} = \sum_{t=0}^T \alpha a_t^{(E)} \delta^t$$

#### 2. Environmental Health

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

#### 3. Equity / Stability

$$f_3^{(C)} = -\text{Var}(X_t)$$

### Interpretation

The community invests in mitigation to balance economic benefits with environmental and intergenerational equity.

---

## (4) Global / Scientific – Robustness and Sustainability Perspective

### Control Variable

$a_t^{(S)}$  : Global policy intervention (offsets, incentives, etc.)

## Deep Uncertainty Set

$$\theta = \{\mu, \sigma, b, q, \delta\} \in \Theta$$

### Objectives

#### 1. Expected performance

$$f_1^{(S)} = \mathbb{E}_\epsilon \left[ \sum_{t=0}^T \alpha a_t^{(E)} \delta^t \right]$$

#### 2. Robustness

$$f_2^{(S)} = \min_{\theta \in \Theta} \left[ \sum_{t=0}^T \alpha a_t^{(E)} \delta^t \right]$$

#### 3. Sustainability

$$f_3^{(S)} = - \max_{\theta \in \Theta} \Pr(X_t \geq X_{\text{crit}}(b, q))$$

### Interpretation

The scientific perspective explores adaptive and robust strategies under deep uncertainty, ensuring resilience across multiple futures.

## 6 Unified System Representation

### State Transition

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

with  $a_t^{(E)} \leq a_t^{(R)}$  and  $X_t < X_{\text{crit}}(b, q)$ .

## 7 Modular Evaluation Framework

Module	Control(s)	Objective(s)	Exogenous Inputs	Evaluated Outputs
Economic Planner	$a_t^{(E)}$	$f^{(E)}$	$a_t^{(R)}, u_t^{(C)}, a_t^{(S)}$	Optimal emission policy
Regulator	$a_t^{(R)}$	$f_1^{(R)}, f_2^{(R)}$	$a_t^{(E)}, u_t^{(C)}, a_t^{(S)}$	Optimal cap policy
Community	$u_t^{(C)}$	$f_1^{(C)}, f_2^{(C)}, f_3^{(C)}$	$a_t^{(E)}, a_t^{(R)}, a_t^{(S)}$	Optimal mitigation strategy
Global / Scientific	$a_t^{(S)}$	$f_1^{(S)}, f_2^{(S)}, f_3^{(S)}$	$a_t^{(E)}, a_t^{(R)}, u_t^{(C)}$	Robust intervention policy

Each module is **self-contained** for evaluation or optimization yet **coupled** through the shared state  $X_t$  and uncertain parameters  $\Theta$ .

## 8 Unified Model Summary (Refined)

$$\begin{aligned}
 X_{t+1} &= X_t + (a_t^{(E)} - u_t^{(C)} + a_t^{(S)}) + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t, \\
 \epsilon_t &\sim \text{LogNormal}(\mu, \sigma^2), \\
 X_t < X_{\text{crit}}(b, q), \quad a_t^{(E)} &\leq a_t^{(R)}, \quad \text{all controls} \geq 0.
 \end{aligned}$$

Each perspective  $i \in \{E, R, C, S\}$  optimizes its own objective function(s)  $f^{(i)}$  under these shared dynamics.

## 9 Conceptual Significance

- ✓ **Consistency:** The regulator's role correctly imposes constraints instead of adding emissions.
- ✓ **Endogeneity:**  $X_{\text{crit}}$  now depends on deep uncertainties  $b$  and  $q$ .
- ✓ **Modularity:** Each stakeholder can be analyzed independently or within a coordinated policy framework.
- ✓ **Realism:** Captures feedbacks between economic, environmental, and social systems under both stochastic and deep uncertainty.

Would you like me to take this refined model and express it formally as a **multi-agent stochastic dynamic game** (where each stakeholder is a player optimizing their own objective given the shared state and exogenous controls)? That would provide a rigorous equilibrium framework for policy analysis.



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.

Perfect — below is a **clean, modular Python implementation** of the **refined unified lake problem model**, designed for extensibility and clarity.

It uses **object-oriented design** to represent each **perspective** as a class with its own control variables and objective evaluation logic, while sharing a **common environment** that simulates lake pollution dynamics.

At the end, a single **interface function** (`simulate_lake_system`) ties everything together: it accepts (1) a dictionary of control variables and (2) a dictionary of uncertain parameters, then returns the **objective values** for all perspectives.

```
python
```

```

import numpy as np

# =====
# Shared Environment Model
# =====

class LakeEnvironment:
    """
    Shared lake environment with pollution dynamics.
    """

    def __init__(self, params):
        """
        params: dictionary containing uncertain parameters and constants
        keys: 'b', 'q', 'mu', 'sigma', 'alpha', 'delta', 'T'
        """
        self.b = params["b"]
        self.q = params["q"]
        self.mu = params["mu"]
        self.sigma = params["sigma"]
        self.alpha = params["alpha"]
        self.delta = params["delta"]
        self.T = params["T"]

        # Derive X_crit dynamically from b and q
        self.X_crit = self._compute_Xcrit()

    def _compute_Xcrit(self):
        """
        Compute X_crit from the implicit condition:
        ( $X^q$ )/(1 +  $X^q$ ) = b * X
        Use a simple numerical root finder.
        """
        from scipy.optimize import brentq

        def f(X):
            return (X ** self.q) / (1 + X ** self.q) - self.b * X

        return brentq(f, 1e-6, 1e3) # safe numerical range

    def step(self, X_t, a_E, u_C, a_S):
        """
        Compute next-period pollution level.
         $X_{t+1} = X_t + (a_E - u_C + a_S) + (X_t^q)/(1 + X_t^q) - b*X_t + \epsilon_t$ 
        """
        epsilon_t = np.random.lognormal(mean=self.mu, sigma=self.sigma)
        X_next = X_t + (a_E - u_C + a_S) + (X_t ** self.q) / (1 + X_t ** self.q) - self.b * X_t
        + epsilon_t
        return X_next

    # =====
    # Perspective 1: Economic Planner
    # =====

    class EconomicPlanner:
        def __init__(self, env):
            self.env = env

        def objective(self, a_E_series):
            """Maximize discounted economic benefit."""
            T = self.env.T
            alpha, delta = self.env.alpha, self.env.delta
            return np.sum([alpha * a_E_series[t] * (delta ** t) for t in range(T)])

    # =====
    # Perspective 2: Environmental Regulator
    # =====

    class EnvironmentalRegulator:
        def __init__(self, env):

```

```

self.env = env

def objectives(self, X_series):
    """
    1. Minimize expected pollution
    2. Minimize eutrophication risk
    """
    mean_pollution = np.mean(X_series)
    risk = np.mean(np.array(X_series) >= self.env.X_crit)
    return {"pollution": mean_pollution, "risk": risk}

# =====
# Perspective 3: Community / Public Health
# =====

class Community:
    def __init__(self, env):
        self.env = env

    def objectives(self, a_E_series, X_series):
        """
        Multi-objective:
        1. Economic well-being (benefit)
        2. Environmental quality (negative pollution)
        3. Equity (negative variance)
        """
        alpha, delta = self.env.alpha, self.env.delta
        T = self.env.T

        f1 = np.sum([alpha * a_E_series[t] * (delta ** t) for t in range(T)])
        f2 = -np.sum([X_series[t] * (delta ** t) for t in range(T)])
        f3 = -np.var(X_series)

        return {"economic": f1, "environment": f2, "equity": f3}

# =====
# Perspective 4: Global / Scientific (Robustness)
# =====

class GlobalScientist:
    def __init__(self, env):
        self.env = env

    def objectives(self, a_E_series, X_series, param_set_list):
        """
        Evaluate robustness under deep uncertainty.
        Each param_set in param_set_list is a dictionary with {b, q, mu, sigma, delta}.
        """
        alpha = self.env.alpha

        # Expected performance
        expected_perf = np.mean([np.sum([alpha * a_E_series[t] * (p["delta"] ** t)
                                         for t in range(self.env.T)]) for p in param_set_list])

        # Worst-case (robustness)
        worst_case = np.min([np.sum([alpha * a_E_series[t] * (p["delta"] ** t)
                                     for t in range(self.env.T)]) for p in param_set_list])

        # Sustainability: average probability of eutrophication across uncertain sets
        risks = []
        for p in param_set_list:
            # recompute X_crit for each parameter set
            b, q = p["b"], p["q"]
            def f(X): return (X ** q) / (1 + X ** q) - b * X
            from scipy.optimize import brentq
            Xcrit_p = brentq(f, 1e-6, 1e3)
            risks.append(np.mean(np.array(X_series) >= Xcrit_p))
        sustainability = -np.max(risks)

        return {"expected": expected_perf, "robustness": worst_case, "sustainability": sustainability}

```

```

# =====
# Unified Simulation Interface
# =====

def simulate_lake_system(control_dict, param_dict):
    """
    Unified simulation function.

    Inputs:
    - control_dict: dictionary with time series or constants for controls
      keys: 'a_E', 'a_R', 'u_C', 'a_S'
    - param_dict: dictionary with uncertain parameter values
      keys: 'b', 'q', 'mu', 'sigma', 'alpha', 'delta', 'T'

    Returns:
    - dict of objective values from all perspectives
    """

    # Initialize environment
    env = LakeEnvironment(param_dict)

    # Unpack controls
    a_E = np.array(control_dict["a_E"])
    a_R = np.array(control_dict["a_R"])
    u_C = np.array(control_dict["u_C"])
    a_S = np.array(control_dict["a_S"])

    # Enforce regulatory cap: a_E <= a_R
    a_E = np.minimum(a_E, a_R)

    # Initialize simulation
    X_series = [0.0]
    X_t = 0.0
    for t in range(env.T):
        X_next = env.step(X_t, a_E[t], u_C[t], a_S[t])
        X_series.append(X_next)
        X_t = X_next

    # Compute objectives
    econ = EconomicPlanner(env).objective(a_E)
    reg = EnvironmentalRegulator(env).objectives(X_series)
    comm = Community(env).objectives(a_E, X_series)

    # Example uncertainty scenarios for the scientist perspective
    theta_list = [
        {"b": param_dict["b"] * 0.9, "q": param_dict["q"], "mu": param_dict["mu"], "sigma": param_dict["sigma"], "delta": param_dict["delta"]},
        {"b": param_dict["b"], "q": param_dict["q"] * 1.1, "mu": param_dict["mu"], "sigma": param_dict["sigma"], "delta": param_dict["delta"]},
        {"b": param_dict["b"], "q": param_dict["q"], "mu": param_dict["mu"] * 1.1, "sigma": param_dict["sigma"], "delta": param_dict["delta"]}
    ]
    sci = GlobalScientist(env).objectives(a_E, X_series, theta_list)

    # Combine results
    results = {
        "EconomicPlanner": {"economic_benefit": econ},
        "Regulator": reg,
        "Community": comm,
        "Scientist": sci,
        "StateTrajectory": X_series
    }

    return results

```

## ✓ Key Features of This Implementation

- Modularity** – Each stakeholder is encapsulated as a class with its own objective logic.
- Shared Environment** – The `LakeEnvironment` governs pollution dynamics and uncertain processes.

3. **Dynamic  $X_{crit}(b, q)$**  – Computed endogenously for each uncertainty configuration.
4. **Regulator constraint enforced** –  $a_t^{(E)} \leq a_t^{(R)}$ .
5. **Unified interface** – `simulate_lake_system()` accepts flexible control and parameter inputs.
6. **Extendable** – The model can easily be expanded for equilibrium simulation, adaptive learning, or multi-objective optimization.

Would you like me to extend this into a **multi-agent simulation** (where each class can iteratively update its controls to respond to others, forming a game-like dynamic)?



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

Excellent — you're absolutely right on both counts.

Let's refine the implementation carefully, preserving modularity and correctness.

## Corrections to Apply

### 1 Bounded Range for $X_{crit}$

The critical threshold  $X_{crit}(b, q)$  must lie in **[0.01, 1.5]**, since the lake's eutrophication threshold is within this physically meaningful range.

So we'll numerically solve

$$\frac{X^q}{1 + X^q} = bX$$

only within **[0.01, 1.5]**, using safe fallback logic if no root is found.

### 2 Correct Lognormal Sampling

In the lake problem, the natural inflow  $\epsilon_t$  is **lognormally distributed with mean  $\mu$  and variance  $\sigma^2$** . However, `np.random.lognormal(mean=μ, sigma=σ)` in NumPy interprets these as the *mean and std of the underlying normal distribution*, not the *mean and std of the lognormal variable*.

We must therefore **convert** the user-specified lognormal parameters  $(\mu, \sigma)$  to the correct underlying normal parameters  $(m, s)$ :

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

Then we sample:

$$\epsilon_t \sim \text{LogNormal}(m, s)$$

## ✓ Refined Implementation

Below is the **corrected and robust Python model**, with both fixes applied and explanatory comments added.

```

python

import numpy as np
from scipy.optimize import brentq

# =====
# Shared Environment Model
# =====

class LakeEnvironment:
    """
    Shared lake environment with pollution dynamics.
    """

    def __init__(self, params):
        """
        params: dictionary containing uncertain parameters and constants
        keys: 'b', 'q', 'mu', 'sigma', 'alpha', 'delta', 'T'
        """
        self.b = params["b"]
        self.q = params["q"]
        self.mu = params["mu"]
        self.sigma = params["sigma"]
        self.alpha = params["alpha"]
        self.delta = params["delta"]
        self.T = params["T"]

        # Derive X_crit dynamically from b and q within [0.01, 1.5]
        self.X_crit = self._compute_Xcrit()

        # Convert given lognormal mean/variance ( $\mu$ ,  $\sigma^2$ ) to normal parameters
        self._set_lognormal_parameters()

    # -----
    # Compute Xcrit from implicit condition:  $(X^q)/(1 + X^q) = b * X$ 
    # -----
    def _compute_Xcrit(self):
        def f(X):
            return (X ** self.q) / (1 + X ** self.q) - self.b * X

        try:
            return brentq(f, 0.01, 1.5) # constrained range
        except ValueError:
            # If no root found in range, use midpoint as fallback
            return 0.75

    # -----
    # Convert lognormal mean/variance to normal parameters
    # -----
    def _set_lognormal_parameters(self):
        mu, sigma = self.mu, self.sigma
        s = np.sqrt(np.log(1 + (sigma ** 2) / (mu ** 2)))
        m = np.log(mu ** 2 / np.sqrt(mu ** 2 + sigma ** 2))
        self.ln_mean = m
        self.ln_sigma = s

    # -----
    # One-step state transition
    # -----
    def step(self, X_t, a_E, u_C, a_S):
        """
        Compute next-period pollution level:
         $X_{t+1} = X_t + (a_E - u_C + a_S) + (X_t^q)/(1 + X_t^q) - b*X_t + \epsilon_t$ 
        """
        epsilon_t = np.random.lognormal(mean=self.ln_mean, sigma=self.ln_sigma)

```

```

        X_next = X_t + (a_E - u_C + a_S) + (X_t ** self.q) / (1 + X_t ** self.q) - self.b * X_t
+ epsilon_t
        return X_next

# =====
# Perspective 1: Economic Planner
# =====

class EconomicPlanner:
    def __init__(self, env):
        self.env = env

    def objective(self, a_E_series):
        """Maximize discounted economic benefit."""
        T = self.env.T
        alpha, delta = self.env.alpha, self.env.delta
        return np.sum([alpha * a_E_series[t] * (delta ** t) for t in range(T)])


# =====
# Perspective 2: Environmental Regulator
# =====

class EnvironmentalRegulator:
    def __init__(self, env):
        self.env = env

    def objectives(self, X_series):
        """
        1. Minimize expected pollution
        2. Minimize eutrophication risk
        """
        mean_pollution = np.mean(X_series)
        risk = np.mean(np.array(X_series) >= self.env.X_crit)
        return {"pollution": mean_pollution, "risk": risk}

# =====
# Perspective 3: Community / Public Health
# =====

class Community:
    def __init__(self, env):
        self.env = env

    def objectives(self, a_E_series, X_series):
        """
        Multi-objective:
        1. Economic well-being
        2. Environmental quality
        3. Equity (variance)
        """
        alpha, delta = self.env.alpha, self.env.delta
        T = self.env.T

        f1 = np.sum([alpha * a_E_series[t] * (delta ** t) for t in range(T)])
        f2 = -np.sum([X_series[t] * (delta ** t) for t in range(T)])
        f3 = -np.var(X_series)

        return {"economic": f1, "environment": f2, "equity": f3}

# =====
# Perspective 4: Global / Scientific (Robustness)
# =====

class GlobalScientist:
    def __init__(self, env):
        self.env = env

    def objectives(self, a_E_series, X_series, param_set_list):
        """
        Evaluate robustness under deep uncertainty.
        Each param_set in param_set_list is a dict with {b, q, mu, sigma, delta}.
        """

```

```

"""
alpha = self.env.alpha
T = self.env.T

# Expected performance
expected_perf = np.mean([
    np.sum([alpha * a_E_series[t] * (p["delta"] ** t) for t in range(T)])
        for p in param_set_list
])

# Worst-case (robustness)
worst_case = np.min([
    np.sum([alpha * a_E_series[t] * (p["delta"] ** t) for t in range(T)])
        for p in param_set_list
])

# Sustainability: average probability of eutrophication across uncertain sets
risks = []
for p in param_set_list:
    b, q = p["b"], p["q"]

    def f(X):
        return (X ** q) / (1 + X ** q) - b * X

    try:
        Xcrit_p = brentq(f, 0.01, 1.5)
    except ValueError:
        Xcrit_p = 0.75 # fallback
    risks.append(np.mean(np.array(X_series) >= Xcrit_p))

sustainability = -np.max(risks)

return {
    "expected": expected_perf,
    "robustness": worst_case,
    "sustainability": sustainability
}

# =====
# Unified Simulation Interface
# =====

def simulate_lake_system(control_dict, param_dict):
    """
    Unified simulation interface.

    Inputs:
    - control_dict: dictionary with control time series
        keys: 'a_E', 'a_R', 'u_C', 'a_S'
    - param_dict: dictionary with uncertain parameter values
        keys: 'b', 'q', 'mu', 'sigma', 'alpha', 'delta', 'T'

    Returns:
    - dict: objectives for each perspective
    """

    # Initialize environment
    env = LakeEnvironment(param_dict)

    # Unpack controls
    a_E = np.array(control_dict["a_E"])
    a_R = np.array(control_dict["a_R"])
    u_C = np.array(control_dict["u_C"])
    a_S = np.array(control_dict["a_S"])

    # Enforce regulatory cap
    a_E = np.minimum(a_E, a_R)

    # Simulate dynamics
    X_series = [0.0]
    X_t = 0.0
    for t in range(env.T):
        X_next = env.step(X_t, a_E[t], u_C[t], a_S[t])
        X_series.append(X_next)

```

```

X_t = X_next

# Compute objectives for all perspectives
econ = EconomicPlanner(env).objective(a_E)
reg = EnvironmentalRegulator(env).objectives(X_series)
comm = Community(env).objectives(a_E, X_series)

# Example uncertainty sets for scientist perspective
theta_list = [
    {"b": param_dict["b"] * 0.9, "q": param_dict["q"], "mu": param_dict["mu"], "sigma": param_dict["sigma"], "delta": param_dict["delta"]},
    {"b": param_dict["b"], "q": param_dict["q"] * 1.1, "mu": param_dict["mu"], "sigma": param_dict["sigma"], "delta": param_dict["delta"]},
    {"b": param_dict["b"], "q": param_dict["q"], "mu": param_dict["mu"] * 1.1, "sigma": param_dict["sigma"], "delta": param_dict["delta"]}
]
sci = GlobalScientist(env).objectives(a_E, X_series, theta_list)

# Combine and return results
return {
    "EconomicPlanner": {"economic_benefit": econ},
    "Regulator": reg,
    "Community": comm,
    "Scientist": sci,
    "StateTrajectory": X_series
}

```

## ✓ Summary of Fixes

Issue	Previous	Fixed
$X_{\text{crit}}$ range	Solved over [1e-6, 1e3]	Restricted to [0.01, 1.5] for realism
Lognormal distribution	Treated $\mu, \sigma$ as normal parameters	Correctly converted $(\mu, \sigma) \rightarrow (m, s)$ for true lognormal sampling
Fallback logic	None	Added robust default $X_{\text{crit}} = 0.75$ if no root found

Would you like me to extend this further so that the **interface function can run deterministic or stochastic ensemble simulations** (e.g., Monte Carlo runs over the stochastic inflows) to compute expected values of objectives under uncertainty?



End here.

Understood ✓ — we'll stop here.

You now have a **fully refined, modular, and internally consistent Python implementation** of the unified lake problem model with correct handling of  $X_{\text{crit}}$  and lognormal uncertainty.