



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

\begin{equation}

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

\end{equation}

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

\begin{equation}

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

\end{equation}

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

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

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

Here's a formalized model of the **Lake Problem**, including all its key components and the mathematical structure used to represent uncertainty and decision-making.

1. Model Overview

The Lake Problem models the tradeoff between **economic benefits** from pollution-generating activities and the **risk of irreversible environmental degradation** (eutrophication) of a lake.

The goal is to choose a pollution emission policy a_t over time that maximizes discounted economic benefits while keeping pollution below a critical threshold X_{crit} .

2. State Variables

Symbol	Description	Domain
X_t	Pollution stock in the lake at time t	$X_t \geq 0$

- Initial condition: $X_0 = 0$
- State evolves over time through the pollution transition function.

3. Decision (Control) Variable

Symbol	Description	Domain
a_t	Anthropogenic pollution emission rate at time t	$a_t \geq 0$

This is the main decision variable under the control of the policymaker.

4. Stochastic Variable

Symbol	Description	Distribution	Parameters
ϵ_t	Natural (stochastic) pollution inflow	Lognormal	$\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$

This captures random natural variations in pollution inflow due to weather, runoff, or other natural events.

5. Constant Parameters

Symbol	Description
α	Benefit-to-pollution ratio (economic return per unit emission)
X_{crit}	Critical pollution threshold beyond which eutrophication becomes irreversible

6. Deep Uncertainty Parameters

These parameters are not known precisely and may vary across plausible scenarios (representing **deep uncertainty**).

Symbol	Description	Typical Role
μ	Mean of natural inflow distribution	Controls average stochastic inflow
σ	Standard deviation of natural inflow	Controls stochastic volatility
b	Natural removal rate of lake	Controls rate of pollutant decay
q	Natural recycling rate	Controls nonlinearity in recycling effect
δ	Discount rate	Governs time preference in benefits

These are uncertain structural parameters — not just random draws, but uncertain *conceptual* quantities whose true values may be unknown or contested.

7. Transition (State Evolution) Function

The pollution dynamics are governed by:

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

where:

- X_t : current pollution stock
 - a_t : anthropogenic emissions
 - $\frac{X_t^q}{1+X_t^q}$: natural recycling process (nonlinear feedback)
 - bX_t : natural removal
 - ϵ_t : stochastic natural inflow
-

8. Objective Function

The policymaker's goal is to maximize the **expected discounted economic benefit**:

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

subject to:

- the pollution dynamics (transition equation),
- $X_t < X_{\text{crit}}$ to avoid eutrophication.

If eutrophication occurs (i.e., $X_t \geq X_{\text{crit}}$), an irreversible penalty or loss can be introduced (e.g., setting future benefits to zero or adding a large cost term).

9. Constraints

$$\begin{aligned} X_{t+1} &= X_t + a_t + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t, \\ X_0 &= 0, \\ X_t &< X_{\text{crit}}, \quad \forall t, \\ a_t &\geq 0. \end{aligned}$$

10. Uncertainty Representation

- **Stochastic uncertainty**: arises from $\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$
- **Deep uncertainty**: arises from uncertain parameters $(\mu, \sigma, b, q, \delta)$

In robust decision-making (RDM) or dynamic programming under uncertainty, the model can be evaluated under multiple plausible parameter sets to test policy robustness.

11. Model Summary

Component	Symbol / Equation	Type
State variable	X_t	Continuous
Decision variable	a_t	Continuous control
Transition function	$X_{t+1} = X_t + a_t + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t$	Dynamic system
Objective	$\max \mathbb{E} [\sum_t \alpha a_t \delta^t]$	Expected utility
Constraints	$X_t < X_{\text{crit}}, a_t \geq 0$	Safety & feasibility
Stochastic variable	$\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$	Random
Deep uncertainties	$\mu, \sigma, b, q, \delta$	Structural

Would you like me to express this in a **dynamic programming (Bellman equation)** form next (for example, showing the optimal policy recursion)? That's a natural next step for formal model development.



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

Excellent observation — you are absolutely correct.

The eutrophication threshold X_{crit} is not fixed; rather, it depends on the lake's **natural removal rate b** and **natural recycling rate q** , both of which are subject to *deep uncertainty*. Let's refine the model accordingly, making X_{crit} an *endogenous function* of uncertain parameters instead of a constant.

Refined Lake Problem Model under Deep Uncertainty

1. State Variable

Symbol	Description	Domain
X_t	Lake pollution stock at time t	$X_t \geq 0$

Represents the current level of pollution in the lake ecosystem.

2. Decision (Control) Variable

Symbol	Description	Domain
a_t	Anthropogenic pollution emission rate at time t	$a_t \geq 0$

Chosen by the decision-maker each year to balance short-term economic gain and long-term environmental sustainability.

3. Stochastic Variable

Symbol	Description	Distribution	Parameters
ϵ_t	Natural stochastic pollution inflow	Lognormal	$\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$

Captures random natural events (e.g., rainfall, runoff, natural inflows).

4. Deep Uncertainty Parameters

Symbol	Description	Role / Interpretation
μ	Mean of lognormal inflow distribution	Average natural pollution level
σ	Standard deviation of inflow distribution	Volatility of natural inflow
b	Natural removal rate of the lake	Controls pollutant decay rate
q	Recycling (nonlinear feedback) rate	Controls self-reinforcing pollution dynamics
δ	Discount rate	Reflects time preference for economic benefits

These are *deeply uncertain*, meaning their true values are not known, agreed upon, or statistically estimable.

Instead, plausible **ranges** or **scenarios** are explored (as in robust decision-making frameworks).

5. Endogenous Critical Threshold $X_{\text{crit}}(b, q)$

The critical threshold occurs when the **natural recycling** of pollutants equals the **natural removal** rate — that is, when the lake ecosystem reaches a tipping point beyond which it cannot recover naturally.

We define X_{crit} implicitly as the solution to:

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

This yields the equilibrium condition where recycling and removal balance.

Since b and q are deeply uncertain, the critical threshold X_{crit} itself is **uncertain** and **scenario-dependent**.

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

where $f(\cdot)$ is derived implicitly from the above nonlinear equilibrium equation.

6. Transition Function

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

where:

- X_t : current lake pollution
- a_t : anthropogenic emission (decision variable)
- $\frac{X_t^q}{1+X_t^q}$: recycling feedback term
- bX_t : natural removal
- ϵ_t : stochastic natural inflow

7. Objective Function

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

subject to:

$$X_{t+1} = X_t + a_t + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t, \quad X_0 = 0,$$

and

$$X_t < X_{\text{crit}}(b, q), \quad a_t \geq 0.$$

8. Uncertainty Representation

Type	Source	Representation
Stochastic uncertainty	Natural inflow ϵ_t	Random draws from $\text{LogNormal}(\mu, \sigma^2)$
Deep uncertainty	Parameters $\mu, \sigma, b, q, \delta$	Represented as uncertain constants with plausible ranges or scenario sets
Derived uncertainty	$X_{\text{crit}}(b, q)$	Computed endogenously from uncertain parameters

9. Reformulated Optimization under Deep Uncertainty

The robust optimization problem can be written as:

$$\max_{\{a_t\}} \min_{\theta \in \Theta} \mathbb{E}_{\epsilon_t | \theta} \left[\sum_t \alpha a_t \delta_\theta^t \right]$$

subject to:

$$X_{t+1} = X_t + a_t + \frac{X_t^{q_\theta}}{1+X_t^{q_\theta}} - b_\theta X_t + \epsilon_t, \quad X_t < X_{\text{crit}}(b_\theta, q_\theta)$$

where:

- $\theta = (\mu, \sigma, b, q, \delta)$ represents a specific *worldview* or *scenario* within the uncertainty set Θ .

This formulation supports robust, adaptive, or exploratory decision analysis methods (e.g., **RDM**, **info-gap**, or **dynamic adaptive policy pathways**).

10. Model Summary (Refined)

Component	Symbol / Equation	Notes
State variable	X_t	Pollution stock
Control variable	a_t	Anthropogenic pollution
Transition function	$X_{t+1} = X_t + a_t + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t$	Nonlinear dynamic system
Objective	$\max \mathbb{E} [\sum_t \alpha a_t \delta^t]$	Expected economic benefit
Critical threshold	$X_{\text{crit}}(b, q) : \frac{X_{\text{crit}}^q}{1+X_{\text{crit}}^q} = bX_{\text{crit}}$	Endogenous tipping point
Constraints	$X_t < X_{\text{crit}}(b, q), a_t \geq 0$	Avoid eutrophication
Stochastic uncertainty	$\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$	Random inflow
Deep uncertainty	$\mu, \sigma, b, q, \delta$	Parameter uncertainty

Would you like me to **derive an explicit analytical or numerical expression** for $X_{\text{crit}}(b, q)$ next (to show how it behaves as b and q vary)? That would make the model operational for simulation or robust policy analysis.



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 moves us into the **multi-stakeholder extension** of the Lake Problem, a key aspect of its interpretation as a **deeply uncertain socio-environmental decision problem**.

Below, I identify **four distinct stakeholder perspectives** that share a **common environment** (the same lake system, same pollution dynamics, same uncertain parameters), but differ in **objectives**, **control variables**, and **decision focus**.

Common Environment (Shared Across Perspectives)

All stakeholders share the same lake ecosystem and biophysical model.

They agree on the following:

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

where:

- X_t : lake pollution at time t
- a_t : anthropogenic pollution emissions (possibly the sum of multiple agents' emissions)
- b : natural removal rate
- q : natural recycling rate
- $\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$: stochastic inflow
- $X_{\text{crit}}(b, q)$: endogenous eutrophication threshold where recycling = removal.

All parameters $\mu, \sigma, b, q, \delta$ are *deeply uncertain* but common to all.

Perspective 1: Economic Developer (Industry / Agriculture Sector)

Motivation

Seeks to **maximize short-term economic benefit** from industrial and agricultural activity that generates pollution. Environmental risks are treated as externalities.

Control Variable

$a_t^{(E)}$: industrial/agricultural pollution emissions at time t

Transition Function

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

where $a_t^{(O)}$ are emissions from other sectors (e.g., households, infrastructure).

Objective Function

$$\max_{\{a_t^{(E)}\}} \mathbb{E} \left[\sum_t \alpha_E a_t^{(E)} \delta^t \right]$$

Constraints

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

Notes

- The developer's focus is **economic benefit** only.
- May assume that environmental collapse is "unlikely" or discounted.

Perspective 2: Environmental Regulator (Government / Environmental Agency)

Motivation

Wants to **ensure environmental safety** by maintaining the lake's health, while allowing limited development.

Control Variable

$u_t^{(R)}$: regulatory limit or pollution cap imposed on total emissions a_t

$$a_t = \min(a_t^{(E)}, u_t^{(R)})$$

Transition Function

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

Objective Functions

The regulator typically has **two conflicting objectives**:

1. **Economic support** (allowing productive activity):

$$f_1^{(R)} = \sum_t \alpha_R a_t \delta^t$$

2. **Environmental protection** (keeping pollution below threshold):

$$f_2^{(R)} = - \sum_t (\max(0, X_t - X_{\text{safe}}))^2$$

or, equivalently, minimizing the expected probability of crossing the threshold:

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

Notes

- This is a **multi-objective optimization** problem (economic + environmental).
- May seek Pareto-optimal policies or safe operating boundaries.



Perspective 3: Local Community / Households

Motivation

Citizens derive both **benefits** from local economic activity (jobs, prosperity) and **disutility** from pollution (health risks, loss of recreation, ecosystem degradation).

Control Variable

$a_t^{(H)}$: household consumption-related pollution or behavioral reduction effort (e.g., conservation practices, wastewater management).

Transition Function

Same shared lake equation:

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

Objective Functions

1. Household economic welfare:

$$f_1^{(H)} = \sum_t \alpha_H a_t^{(E)} \delta^t$$

(they indirectly benefit from industrial productivity)

2. Environmental and health quality of life:

$$f_2^{(H)} = - \sum_t \beta_H X_t \delta^t$$

where β_H captures sensitivity to pollution.

Notes

- This is a **bi-objective model** balancing prosperity and wellbeing.
- $a_t^{(H)}$ could also represent voluntary or subsidized pollution reduction efforts.

Perspective 4: Future Generations / Sustainability Advocate

Motivation

Focuses on **long-term lake resilience** and avoiding irreversible eutrophication, possibly over an infinite or intergenerational horizon.

Control Variable

$a_t^{(S)}$: sustainable emission policy decision (e.g., long-term emission limit or adaptive threshold control)

Transition Function

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

Objective Functions

1. Intergenerational equity / preservation:

$$f_1^{(S)} = - \sum_t \Pr(X_t \geq X_{\text{crit}}(b, q))$$

or equivalently minimize the long-term expected pollution:

$$f_1^{(S)} = -\mathbb{E} \left[\limsup_{t \rightarrow \infty} X_t \right]$$

2. Economic sustainability:

$$f_2^{(S)} = \sum_t \alpha_S a_t^{(S)} \delta^t$$

Notes

- Deeply uncertain parameters b, q, δ are critical because they shape whether recovery is possible.
- The sustainable planner operates with a **precautionary principle** under deep uncertainty.

Summary: Four Stakeholder Perspectives

Perspective	Control Variable	Objectives	Key Focus
1. Economic Developer	$a_t^{(E)}$	$f^{(E)} = \max \sum \alpha_E a_t^{(E)} \delta^t$	Short-term profit
2. Environmental Regulator	$u_t^{(R)}$ (emission cap)	$f_1^{(R)} = \sum \alpha_R a_t \delta^t, f_2^{(R)} = -\Pr(X_t \geq X_{\text{crit}})$	Balanced economy & safety
3. Local Community	$a_t^{(H)}$ (behavioral / consumption)	$f_1^{(H)} = \sum \alpha_H a_t^{(E)} \delta^t, f_2^{(H)} = -\sum \beta_H X_t \delta^t$	Quality of life
4. Future Generations / Sustainability Advocate	$a_t^{(S)}$	$f_1^{(S)} = -\Pr(X_t \geq X_{\text{crit}}), f_2^{(S)} = \sum \alpha_S a_t^{(S)} \delta^t$	Long-term resilience

Shared Uncertainty Structure

All four share:

- Common pollution state dynamics.
- Common deep uncertainties: $\mu, \sigma, b, q, \delta$.
- Shared stochastic disturbance ϵ_t .

Differences lie in:

- Control scope (direct or indirect influence on a_t)
- Time horizon (short vs. long-term)
- Objective emphasis (economic vs. environmental welfare)

Would you like me to **recast these four perspectives in a multi-agent dynamic game formulation** (where each stakeholder's decisions jointly affect X_t) next?

That's often the next step in multi-stakeholder lake modeling (e.g., in dynamic games or cooperative bargaining frameworks).



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 exactly the right next step.

What you're asking for is a **unified, modular multi-perspective Lake Problem model** that ensures all perspectives operate within the **same physical-environmental system**, while preserving each stakeholder's **distinct controls** and **objectives**.

The unified structure should be internally consistent (shared dynamics, shared uncertainties), yet flexible enough that **each perspective can be evaluated independently** by treating others' decisions as exogenous inputs.

Below is the full formalization.

Unified Multi-Perspective Lake Model

1. Common Environment (Shared Across All Perspectives)

State Variable

Symbol	Description	Domain
X_t	Lake pollution stock at time t	$X_t \geq 0$

Represents the cumulative pollution in the lake ecosystem, shared by all actors.

Stochastic Component

Symbol	Description	Distribution
ϵ_t	Natural stochastic pollution inflow	$\text{LogNormal}(\mu, \sigma^2)$

Deeply Uncertain Parameters

All perspectives share the same deeply uncertain environmental and socio-economic parameters:

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

Symbol	Meaning	Type
μ, σ	Mean and standard deviation of natural inflow	stochastic / deep uncertainty
b	Natural removal rate	deep uncertainty

Symbol	Meaning	Type
q	Recycling rate	deep uncertainty
δ	Discount rate	deep uncertainty

Critical Threshold (Endogenous)

The eutrophication threshold depends on the uncertain environmental parameters:

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

The function $f(b, q)$ defines the pollution level where natural recycling equals natural removal.

2. Aggregate Pollution Transition Function

Pollution evolves according to contributions from all stakeholders plus the natural system:

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

where the **total anthropogenic emission** is:

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

and $a_t^{(E)}, a_t^{(H)}, a_t^{(S)}$ denote the respective emissions (or controlled activities) of each stakeholder.
The regulator's policy $u_t^{(R)}$ may **constrain** these emissions:

$$a_t^{(i)} \leq u_t^{(R)}, \quad i \in \{E, H, S\}.$$

3. Shared Constraints

$$\begin{aligned} X_0 &= 0, \\ X_t &< X_{\text{crit}}(b, q) \quad \forall t, \\ a_t^{(E)}, a_t^{(H)}, a_t^{(S)} &\geq 0, \\ u_t^{(R)} &\geq 0. \end{aligned}$$

4. Perspective-Specific Models (Embedded Modules)

Each stakeholder's submodel has:

1. its own **control variables**;
2. its own **objectives**;
3. the **shared state equation**;

4. other stakeholders' controls treated as *exogenous inputs*.

Perspective 1: Economic Developer (Industry / Agriculture)

Control:

$a_t^{(E)}$ — emissions from production and agriculture.

Transition:

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

where $a_t^{(-E)} = a_t^{(H)} + a_t^{(S)}$ are exogenous inputs.

Objective:

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

Constraints:

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

Perspective 2: Environmental Regulator

Control:

$u_t^{(R)}$ — maximum allowed emission level per time period (emission cap).

Transition:

Same shared dynamics, but actual emissions are capped:

$$a_t^{(i)} = \min(a_t^{(i)}, u_t^{(R)}), \quad i \in \{E, H, S\}$$

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

Objectives (multi-objective):

1. Economic productivity support

$$f_1^{(R)} = \sum_t \alpha_R \sum_i a_t^{(i)} \delta^t$$

2. Environmental protection

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

or equivalently minimizing expected threshold exceedance.

Constraints:

$$u_t^{(R)} \geq 0$$

 **Perspective 3: Local Community / Households**
Control:

$a_t^{(H)}$ — household or local emission decision (e.g., lifestyle, conservation).

Transition:

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

where $a_t^{(-H)} = a_t^{(E)} + a_t^{(S)}$.

Objectives (bi-objective):
1. Economic well-being (linked to local productivity):

$$f_1^{(H)} = \sum_t \alpha_H a_t^{(E)} \delta^t$$

(households benefit from industrial activity)

2. Health and environmental quality:

$$f_2^{(H)} = - \sum_t \beta_H X_t \delta^t$$

Constraints:

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

 **Perspective 4: Sustainability Advocate / Future Generations**
Control:

$a_t^{(S)}$ — sustainable emission rate or adaptive mitigation action.

Transition:

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

where $a_t^{(-S)} = a_t^{(E)} + a_t^{(H)}$.

Objectives (multi-objective):
1. Ecosystem preservation:

$$f_1^{(S)} = - \Pr(X_t \geq X_{\text{crit}}(b, q))$$

or equivalently minimizing long-term expected pollution:

$$f_1^{(S)} = -\mathbb{E} \left[\limsup_{t \rightarrow \infty} X_t \right]$$

2. Intergenerational welfare:

$$f_2^{(S)} = \sum_t \alpha_S a_t^{(S)} \delta^t$$

Constraints:

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

5. Unified Model Schema

The unified model can be represented compactly as:

$$\begin{cases} X_{t+1} = X_t + \sum_{i \in \mathcal{I}} a_t^{(i)} + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t, \\ X_0 = 0, \quad X_t < X_{\text{crit}}(b, q), \\ a_t^{(i)} \geq 0, \quad i \in \mathcal{I}, \\ a_t^{(i)} \leq u_t^{(R)} \text{ if regulated.} \end{cases}$$

where the stakeholder set is:

$$\mathcal{I} = \{E, H, S\}.$$

Each stakeholder i solves:

$$\max_{a_t^{(i)}} f_j^{(i)}(a_t^{(i)}, a_t^{(-i)}, X_t, \theta)$$

subject to the shared transition function, with other controls $a_t^{(-i)}$ treated as *exogenous*.

6. Modular Evaluation Framework

This unified structure allows **modular evaluation**:

- Evaluate each perspective individually by fixing other stakeholders' actions $a_t^{(-i)}$.
- Combine perspectives for multi-agent analysis (e.g., coordination, Nash equilibrium, or cooperative Pareto analysis).
- Use the same shared simulation or uncertainty propagation engine for all modules (since X_t , ϵ_t , and θ are common).

7. Summary Table

Component	Shared / Specific	Description
X_t	Shared	Lake pollution state

Component	Shared / Specific	Description
ϵ_t	Shared	Stochastic natural inflow
$\theta = (\mu, \sigma, b, q, \delta)$	Shared	Deeply uncertain parameters
$X_{\text{crit}}(b, q)$	Shared (derived)	Eutrophication threshold
$a_t^{(E)}, a_t^{(H)}, a_t^{(S)}$	Specific	Control variables for each perspective
$u_t^{(R)}$	Specific	Regulator's emission cap
$f^{(E)}, (f_1^{(R)}, f_2^{(R)}), (f_1^{(H)}, f_2^{(H)}), (f_1^{(S)}, f_2^{(S)})$	Specific	Objective functions
Transition Function	Shared	$X_{t+1} = X_t + \sum_i a_t^{(i)} + \frac{X_t^q}{1+X_t^q} - bX_t + \epsilon_t$
Constraints	Shared	Nonnegativity, threshold, and regulation

✓ Result

This **unified multi-perspective lake model**:

- Shares **one consistent environment** and **one state variable** X_t .
- Embeds **perspective-specific controls and objectives** modularly.
- Enables **independent or joint evaluation** of each perspective (treating others as exogenous).
- Retains **deep and stochastic uncertainty** as a common driver of risk and variability.

Would you like me to next show this as a **computational framework or algorithmic structure** (e.g., a pseudo-code or dynamic simulation framework for evaluating each perspective under uncertainty)? This can make it immediately ready for implementation in Python or Julia.



Please refine your aggregated anthropogenic emissions to make them more reasonable. For example, should the control variable for environmental regulators be the limits for each type of emission or the total amount of emissions?

Excellent — this is a very insightful and necessary refinement.

You are absolutely right: in the previously unified model, the **aggregated anthropogenic emissions** term

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

was too coarse, and the **regulator's control** $u_t^{(R)}$ needed a clearer interpretation.

Let's now **refine the model** to make the emission structure and regulatory mechanisms more realistic, internally consistent, and analytically modular — while still maintaining the shared environmental state and uncertainties.

🌐 Refined Unified Multi-Perspective Lake Model

1. Shared Environmental System

State Variable

$$X_t \geq 0$$

Pollution stock in the lake at time t .

Environmental Transition Function

The lake evolves as:

$$X_{t+1} = X_t + \sum_{i \in \{E, H, S\}} e_t^{(i)} + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

where:

- $e_t^{(i)}$: **effective emissions** released into the lake by perspective i .
- $\epsilon_t \sim \text{LogNormal}(\mu, \sigma^2)$: stochastic natural inflow.
- $b, q, \mu, \sigma, \delta$: deeply uncertain parameters (shared).

2. Emission Control Structure

We now distinguish **decision variables** (desired actions) from **effective emissions** (actual realized pollution).

Stakeholder Emission Controls

Each emitting agent $i \in \{E, H, S\}$ chooses a desired emission level:

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

These are the **decision variables** reflecting economic activity or consumption.

Regulatory Control

The **environmental regulator** does *not* emit directly.

Its control variable is now more precisely defined as:

$$u_t^{(R,i)} \quad \text{for } i \in \{E, H, S\}$$

that is, **per-sector emission limits**, not a single total cap.

Each $u_t^{(R,i)}$ defines the maximum allowable emissions for that type of actor.

Hence, **effective emissions** are determined by:

$$e_t^{(i)} = \min(a_t^{(i)}, u_t^{(R,i)})$$

This design provides two major advantages:

- It allows differentiated policies (e.g., stricter industrial caps than household caps).
- It ensures modular consistency (each agent's emissions remain bounded but exogenous to others).

3. Shared Constraints

$$\begin{aligned} X_0 &= 0, \\ X_t &< X_{\text{crit}}(b, q), \\ a_t^{(i)}, u_t^{(R,i)} &\geq 0. \end{aligned}$$

and $X_{\text{crit}}(b, q)$ satisfies:

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

4. Perspective-Specific Submodels (Modular)

Each perspective interacts with the shared environment but optimizes its own objective, treating other agents' controls as exogenous.

(1) Economic Developer (Industry / Agriculture)

Control Variable:

$a_t^{(E)}$: planned industrial/agricultural emissions (e.g., fertilizer use, factory discharges)

Effective Emission:

$$e_t^{(E)} = \min(a_t^{(E)}, u_t^{(R,E)})$$

Transition Function:

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

Objective:

$$\max_{\{a_t^{(E)}\}} f^{(E)} = \mathbb{E} \left[\sum_t \alpha_E e_t^{(E)} \delta^t \right]$$

subject to $a_t^{(E)} \geq 0, X_t < X_{\text{crit}}(b, q)$.

(2) Environmental Regulator

Control Variables:

$$u_t^{(R,E)}, u_t^{(R,H)}, u_t^{(R,S)}$$

(sector-specific emission limits)

Transition Function:

$$X_{t+1} = X_t + \sum_{i \in \{E,H,S\}} \min(a_t^{(i)}, u_t^{(R,i)}) + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

Objectives (multi-objective):

1. Economic support:

$$f_1^{(R)} = \sum_t \alpha_R \sum_i e_t^{(i)} \delta^t$$

2. Environmental protection:

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

or equivalently, minimizing expected exceedance probability or long-term pollution.

(3) Local Community / Households

Control Variable:

$a_t^{(H)}$: planned household emissions (energy use, waste disposal, etc.)

Effective Emission:

$$e_t^{(H)} = \min(a_t^{(H)}, u_t^{(R,H)})$$

Transition Function:

Same shared system.

Objectives (bi-objective):

1. Economic benefit (linked to industrial activity):

$$f_1^{(H)} = \sum_t \alpha_H e_t^{(E)} \delta^t$$

2. Health and environmental quality:

$$f_2^{(H)} = - \sum_t \beta_H X_t \delta^t$$

(4) Sustainability Advocate / Future Generations

Control Variable:

$a_t^{(S)}$: planned sustainable mitigation or emission offset (could be negative if sequestration).

Effective Emission:

$$e_t^{(S)} = \min(a_t^{(S)}, u_t^{(R,S)})$$

Transition Function:

Same shared environmental equation.

Objectives (bi-objective):

1. Long-term ecological resilience:

$$f_1^{(S)} = -\mathbb{E} \left[\limsup_{t \rightarrow \infty} X_t \right]$$

or equivalently, minimizing the chance of eutrophication:

$$f_1^{(S)} = -\Pr(X_t \geq X_{\text{crit}}(b, q))$$

2. Sustainable economic utility:

$$f_2^{(S)} = \sum_t \alpha_S e_t^{(S)} \delta^t$$

5. Final Unified System Summary

Symbol	Description	Controlled by	Type
X_t	Lake pollution state	Shared	State variable
ϵ_t	Natural inflow	Nature	Stochastic variable
$a_t^{(i)}$	Desired emissions or mitigation	Stakeholder i	Control
$u_t^{(R,i)}$	Emission limit for sector i	Regulator	Control
$e_t^{(i)} = \min(a_t^{(i)}, u_t^{(R,i)})$	Effective emission	Derived	Function of both
$b, q, \mu, \sigma, \delta$	Environmental & economic parameters	Shared	Deeply uncertain constants
$X_{\text{crit}}(b, q)$	Eutrophication threshold	Derived	Function of uncertain parameters

Unified Transition Function (Final Form)

$$X_{t+1} = X_t + \sum_{i \in \{E, H, S\}} \min(a_t^{(i)}, u_t^{(R,i)}) + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

Unified Framework Properties

Property	Description
Shared dynamics	All stakeholders influence the same environmental state X_t .
Modular evaluation	Each perspective can be optimized independently by treating others' controls as fixed inputs.

Property	Description
Realistic regulation	Regulator now sets sector-specific emission caps , consistent with policy practice.
Endogenous emissions	Actual pollution $e_t^{(i)}$ depends on both stakeholder intent and regulation.
Deep & stochastic uncertainty	Propagated consistently through all models.

✓ Interpretation

This refinement yields a **modular yet integrated socio-environmental system** where:

- All actors share one environment and uncertainty structure.
- Each actor's behavior and control have a clear physical manifestation.
- The regulator's influence is explicit and differentiated across emission sources.
- The model supports both **independent analysis** (per perspective) and **joint multi-agent evaluation** (e.g., cooperative, competitive, or game-theoretic settings).

Would you like me to now **extend this unified system into a game-theoretic dynamic model**, showing the equilibrium relationships between these actors (e.g., Nash vs. cooperative outcomes)? This is typically the next formal step in multi-perspective lake modeling frameworks.



You treat the control variable of sustainability advocate as an additive contributor to total emissions and assign a emission limit to it, but it should be planned sustainable mitigation or emission offset. Please refine your specification.

Excellent — and you are absolutely right again.

If the **Sustainability Advocate's control** represents **mitigation or offset**, it should *reduce* net emissions, not add to them, and it should be subject to **capacity limits** (e.g., mitigation cost, feasible sequestration), not *emission limits*.

Let's refine the unified model once more to capture this properly, while maintaining a consistent shared structure across all perspectives.

We will:

- preserve the same common environment and uncertainty structure;
- redefine total effective emissions to *subtract* mitigation offsets;
- clarify the regulator's and sustainability advocate's distinct roles.

🌐 Refined Unified Multi-Perspective Lake Model (with Mitigation / Offset)

1. Shared Environmental State and Parameters

State Variable

$$X_t \geq 0$$

Lake pollution stock at time t .

Stochastic Component

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

Random natural inflow of pollution.

Deeply Uncertain Parameters

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

where

- b : natural removal rate,
 - q : natural recycling rate,
 - μ, σ : parameters of stochastic inflow,
 - δ : discount rate.
-

Critical Threshold

Endogenously defined by the balance between recycling and removal:

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

2. Emission–Mitigation Balance (Shared Transition Function)

Pollution evolves as follows:

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

where

- E_t : **aggregate anthropogenic emissions** from economic and household activities,
- M_t : **aggregate mitigation or offset** achieved through sustainability efforts.

Each term is defined below.

Aggregate Anthropogenic Emissions

$$E_t = e_t^{(E)} + e_t^{(H)}$$

with

$$e_t^{(i)} = \min(a_t^{(i)}, u_t^{(R,i)}), \quad i \in \{E, H\}$$

where:

- $a_t^{(i)}$: desired emissions (economic decisions of each actor);
- $u_t^{(R,i)}$: regulator-imposed emission caps for sector i .

Aggregate Mitigation / Offset

$$M_t = m_t^{(S)}$$

with

$$m_t^{(S)} = \min(a_t^{(S)}, c_t^{(S)})$$

where:

- $a_t^{(S)}$: planned mitigation/offset effort (decision of the sustainability advocate),
- $c_t^{(S)}$: **capacity limit** for sustainable mitigation (physical, financial, or technological constraint).

Thus, mitigation acts as a **negative contribution** to total net emissions.

Final Pollution Dynamics (Unified)

$$X_{t+1} = X_t + \left[\sum_{i \in \{E, H\}} \min(a_t^{(i)}, u_t^{(R,i)}) \right] - \min(a_t^{(S)}, c_t^{(S)}) + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

3. Shared Constraints

$$\begin{aligned} X_0 &= 0, \\ X_t &< X_{\text{crit}}(b, q), \\ a_t^{(E)}, a_t^{(H)}, a_t^{(S)}, u_t^{(R,i)}, c_t^{(S)} &\geq 0. \end{aligned}$$

4. Perspective-Specific Submodels

Each stakeholder interacts with the shared dynamics above, treating others' controls as exogenous.

(1) Economic Developer

Control variable: $a_t^{(E)}$ — desired industrial/agricultural emissions.

Effective emissions: $e_t^{(E)} = \min(a_t^{(E)}, u_t^{(R,E)})$

Transition: Shared.

Objective:

$$\max_{\{a_t^{(E)}\}} f^{(E)} = \mathbb{E} \left[\sum_t \alpha_E e_t^{(E)} \delta^t \right]$$

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

(2) Environmental Regulator

Control variables: sector-specific emission caps $u_t^{(R,E)}, u_t^{(R,H)}$.

Transition: Shared dynamics, where $u_t^{(R,i)}$ directly constrain emissions.

Objectives (multi-objective):

1. Economic support

$$f_1^{(R)} = \sum_t \alpha_R (e_t^{(E)} + e_t^{(H)}) \delta^t$$

2. Environmental protection

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

or minimizing long-term pollution.

(3) Local Community / Households

Control variable: $a_t^{(H)}$ — planned household emissions or reductions.

Effective emissions: $e_t^{(H)} = \min(a_t^{(H)}, u_t^{(R,H)})$

Objectives (bi-objective):

1. Economic well-being (benefit from productivity):

$$f_1^{(H)} = \sum_t \alpha_H e_t^{(E)} \delta^t$$

2. Health and environmental quality:

$$f_2^{(H)} = - \sum_t \beta_H X_t \delta^t$$

(4) Sustainability Advocate / Future Generations

Control variable: $a_t^{(S)}$ — planned mitigation or offset investment.

Effective mitigation: $m_t^{(S)} = \min(a_t^{(S)}, c_t^{(S)})$

Transition: Shared dynamics (mitigation enters with a negative sign).

Objectives (multi-objective):

1. Long-term ecological preservation

$$f_1^{(S)} = -\mathbb{E} \left[\limsup_{t \rightarrow \infty} X_t \right]$$

or minimizing the probability of eutrophication:

$$f_1^{(S)} = -\Pr(X_t \geq X_{\text{crit}}(b, q))$$

2. Sustainable economic utility (cost-effectiveness of mitigation)

$$f_2^{(S)} = -\sum_t \gamma_S m_t^{(S)} \delta^t$$

where γ_S represents the cost or resource burden of mitigation.

5. Final Unified Specification

Shared Environmental Dynamics

$$X_{t+1} = X_t + \sum_{i \in \{E, H\}} \min(a_t^{(i)}, u_t^{(R,i)}) - \min(a_t^{(S)}, c_t^{(S)}) + \frac{X_t^q}{1 + X_t^q} - bX_t + \epsilon_t$$

Stakeholder Controls

Actor	Controls	Interpretation
Economic Developer	$a_t^{(E)}$	Production emissions
Households	$a_t^{(H)}$	Lifestyle emissions
Sustainability Advocate	$a_t^{(S)}$	Mitigation / offset actions
Regulator	$u_t^{(R,E)}, u_t^{(R,H)}$	Sectoral emission limits
(System parameter)	$c_t^{(S)}$	Mitigation capacity limit

Uncertainties

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

common to all.

6. Key Improvements and Interpretation

Aspect	Previous Formulation	Refined Formulation
Sustainability control	Treated as additive emission	Treated as mitigation (subtractive effect)
Regulator control	Single total emission cap	Sector-specific emission limits $u_t^{(R,i)}$
Mitigation limits	Treated as emission limit	Explicit mitigation capacity $c_t^{(S)}$
Net pollution balance	$A_t = \sum a_t^{(i)}$	$E_t - M_t = e_t^{(E)} + e_t^{(H)} - m_t^{(S)}$
Consistency	Ambiguous signs	Correct physical direction (mitigation reduces pollution)

Resulting Model

This final specification achieves:

1. **Shared environment and uncertainty structure** across all perspectives.
2. **Clear physical interpretation** — mitigation reduces total emissions.
3. **Modular design** — each perspective can be evaluated independently (others' controls exogenous).
4. **Realistic regulation** — emission limits apply to emitters only.
5. **Integrated offset mechanism** — sustainability actions explicitly mitigate pollution within the same system.

Would you like me to now **add a social planner or coordination layer** that combines these four modules (e.g., exploring cooperative vs. non-cooperative outcomes or Pareto frontiers between objectives)? This would complete the system as a multi-actor decision environment.



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 — we'll now implement a **modular, object-oriented Python structure** for the refined unified Lake Problem model.

This implementation will:

- use **classes** to represent each perspective (Economic Developer, Regulator, Local Community, Sustainability Advocate);
- include a **shared environment class** for the lake system;
- support deep uncertainty via parameter inputs;
- expose a single **interface function** that takes control inputs and uncertain parameters, simulates the dynamics, and returns all objective values.

python

```

# =====
# Unified Modular Lake Problem Model (Refined)
# =====
# This code defines classes for different stakeholder perspectives
# that share a common lake environment. Each actor has its own
# control variables and objectives, and all interact through the
# shared pollution state X_t.
# =====

import numpy as np

# -----
# Shared Environment Class
#
class LakeEnvironment:
    def __init__(self, params, T=100):
        """
        params: dictionary of uncertain parameters
            keys: ['mu', 'sigma', 'b', 'q', 'delta']
        T: simulation time horizon
        """
        self.mu = params.get('mu', 0.02)
        self.sigma = params.get('sigma', 0.01)
        self.b = params.get('b', 0.42)
        self.q = params.get('q', 2.0)
        self.delta = params.get('delta', 0.98)
        self.T = T
        self.X = np.zeros(T + 1) # pollution stock over time
        self.eps = np.random.lognormal(mean=self.mu, sigma=self.sigma, size=T)

    def X_crit(self):
        """Compute pollution threshold X_crit where recycling = removal"""
        # Solve for X_crit numerically
        from scipy.optimize import fsolve
        f = lambda X: (X**self.q) / (1 + X**self.q) - self.b * X
        Xc = fsolve(f, 1.0)[0]
        return Xc

    def update(self, emissions, mitigation, t):
        """Update lake pollution based on total emissions and mitigation"""
        Xt = self.X[t]
        next_X = Xt + emissions - mitigation + (Xt**self.q) / (1 + Xt**self.q) - self.b * Xt + self.eps[t]
        self.X[t + 1] = max(next_X, 0) # ensure nonnegative
        return self.X[t + 1]

# -----
# Stakeholder Base Class
#
class Stakeholder:
    def __init__(self, name, delta):
        self.name = name
        self.delta = delta
        self.objectives = {}

    def compute_discount(self, t):
        return self.delta ** t

# -----
# Economic Developer (Industry / Agriculture)
#
class EconomicDeveloper(Stakeholder):
    def __init__(self, alpha_E, delta):
        super().__init__("Economic Developer", delta)
        self.alpha_E = alpha_E

    def effective_emission(self, a_E, u_E):
        return min(a_E, u_E)

```

```

def objective(self, emissions):
    """Compute discounted economic benefit"""
    total = sum(self.alpha_E * e * self.compute_discount(t) for t, e in
enumerate(emissions))
    self.objectives["economic_benefit"] = total
    return self.objectives

# -----
# Environmental Regulator
# -----
class EnvironmentalRegulator(Stakeholder):
    def __init__(self, alpha_R, delta):
        super().__init__("Environmental Regulator", delta)
        self.alpha_R = alpha_R

    def objectives_fn(self, emissions, X, Xcrit):
        """Two objectives: economic support and environmental protection"""
        econ_support = sum(self.alpha_R * sum(e) * self.compute_discount(t) for t, e in
enumerate(emissions))
        env_protect = -np.mean(X >= Xcrit)
        self.objectives = {
            "economic_support": econ_support,
            "environmental_protection": env_protect,
        }
    return self.objectives

# -----
# Local Community / Households
# -----
class LocalCommunity(Stakeholder):
    def __init__(self, alpha_H, beta_H, delta):
        super().__init__("Local Community", delta)
        self.alpha_H = alpha_H
        self.beta_H = beta_H

    def objectives_fn(self, industrial_emissions, X):
        """Two objectives: economic well-being and health/environmental quality"""
        econ_benefit = sum(self.alpha_H * e * self.compute_discount(t) for t, e in
enumerate(industrial_emissions))
        health_quality = -sum(self.beta_H * x * self.compute_discount(t) for t, x in
enumerate(X))
        self.objectives = {
            "economic_wellbeing": econ_benefit,
            "environmental_quality": health_quality,
        }
    return self.objectives

# -----
# Sustainability Advocate / Future Generations
# -----
class SustainabilityAdvocate(Stakeholder):
    def __init__(self, gamma_S, delta):
        super().__init__("Sustainability Advocate", delta)
        self.gamma_S = gamma_S

    def effective_mitigation(self, a_S, c_S):
        return min(a_S, c_S)

    def objectives_fn(self, X, mitigation, Xcrit):
        """Two objectives: long-term preservation and mitigation cost"""
        preserve = -np.mean(X >= Xcrit)
        cost = -sum(self.gamma_S * m * self.compute_discount(t) for t, m in
enumerate(mitigation))
        self.objectives = {
            "ecological_preservation": preserve,
            "mitigation_cost": cost,
        }
    return self.objectives

```

```

# Unified Simulation Interface
#
def simulate_lake_system(controls, params, T=100):
    """
    Unified interface to simulate the lake problem.
    Inputs:
        controls: dict containing time series or scalars for control variables:
        {
            'a_E': [...],      # planned emissions by Economic Developer
            'a_H': [...],      # planned emissions by Households
            'a_S': [...],      # planned mitigation by Sustainability Advocate
            'u_R_E': [...],    # emission cap for industry
            'u_R_H': [...],    # emission cap for households
            'c_S': [...]       # mitigation capacity limit
        }

        params: dict of uncertain parameters:
        {
            'mu': ..., 'sigma': ..., 'b': ..., 'q': ..., 'delta': ...,
            'alpha_E': ..., 'alpha_R': ..., 'alpha_H': ..., 'beta_H': ..., 'gamma_S': ...
        }

    Output:
        dict of all objective function values for each stakeholder.
    """

# --- initialize environment ---
env = LakeEnvironment(params, T)
Xcrit = env.X_crit()

# --- initialize stakeholders ---
dev = EconomicDeveloper(params['alpha_E'], env.delta)
reg = EnvironmentalRegulator(params['alpha_R'], env.delta)
com = LocalCommunity(params['alpha_H'], params['beta_H'], env.delta)
sus = SustainabilityAdvocate(params['gamma_S'], env.delta)

# --- ensure controls are arrays over time ---
def arr(key):
    val = controls.get(key, 0)
    return np.array(val if isinstance(val, (list, np.ndarray)) else [val] * T)

a_E, a_H, a_S = arr('a_E'), arr('a_H'), arr('a_S')
u_R_E, u_R_H = arr('u_R_E'), arr('u_R_H')
c_S = arr('c_S')

# --- time loop ---
e_E, e_H, m_S = [], [], []

for t in range(T):
    # effective emissions and mitigation
    eE = dev.effective_emission(a_E[t], u_R_E[t])
    eH = min(a_H[t], u_R_H[t])
    mS = sus.effective_mitigation(a_S[t], c_S[t])

    # update environment
    env.update(eE + eH, mS, t)

    e_E.append(eE)
    e_H.append(eH)
    m_S.append(mS)

# --- compute objectives ---
obj_dev = dev.objective(e_E)
obj_reg = reg.objectives_fn(list(zip(e_E, e_H)), env.X, Xcrit)
obj_com = com.objectives_fn(e_E, env.X)
obj_sus = sus.objectives_fn(env.X, m_S, Xcrit)

# --- return results ---
return {
    "EconomicDeveloper": obj_dev,
    "EnvironmentalRegulator": obj_reg,
    "LocalCommunity": obj_com,
    "SustainabilityAdvocate": obj_sus,
}

```

```
# =====
# Example usage (not executed):
# results = simulate_lake_system(control_dict, param_dict)
# =====
```

✓ Implementation Highlights

- **Classes for modularity:** Each stakeholder class encapsulates its control logic and objectives.
- **Shared dynamics:** `LakeEnvironment` contains the state evolution and uncertainty.
- **Clear interfaces:** `simulate_lake_system()` acts as the unified model driver.
- **Extensible:** You can add coordination, optimization, or uncertainty sampling easily.
- **Output:** A structured dictionary returning all objective values, e.g.:

```
python

{
    "EconomicDeveloper": {"economic_benefit": ...},
    "EnvironmentalRegulator": {"economic_support": ..., "environmental_protection": ...},
    "LocalCommunity": {"economic_wellbeing": ..., "environmental_quality": ...},
    "SustainabilityAdvocate": {"ecological_preservation": ..., "mitigation_cost": ...}
}
```

Would you like me to extend this into a **multi-scenario evaluation framework** (e.g., running `simulate_lake_system` over a set of uncertain parameter samples to estimate robustness or regret)?



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

Excellent points — both critiques are entirely correct.

Let's fix them properly and refine the model implementation to be **numerically consistent** and **physically realistic**.

✓ Refinements to Implement

1. Improving the computation of X_{crit} :

The eutrophication threshold should be found using a **bounded root-finding method** (e.g. `brentq`), ensuring that

$$X_{\text{crit}} \in [0.01, 1.5].$$

This ensures numerical stability and realistic results even under deep uncertainty.

2. Correcting the lognormal inflow generation:

In Python, `np.random.lognormal(mean, sigma)` interprets the parameters as the **mean and standard**

deviation of the underlying normal distribution, *not* the mean and variance of the lognormal distribution*.

To model the inflow with a lognormal distribution that has *mean* = μ and *variance* = σ^2 in the **original space**, we must first convert those to the equivalent normal parameters:

$$\phi = \sqrt{\ln(1 + (\sigma^2/\mu^2))},$$

$$\psi = \ln(\mu) - \frac{1}{2}\phi^2.$$

Then we can sample as:

```
python
np.random.lognormal(mean=ψ, sigma=φ)
```

Refined Python Implementation

Below is the updated version with these fixes applied.

```
python
# -----
# Refined Unified Modular Lake Problem Model
# -----
# - Corrects X_crit root finding within [0.01, 1.5]
# - Corrects lognormal sampling for mean μ and variance σ²
# -----

import numpy as np
from scipy.optimize import brentq

#
# Shared Environment Class
#
class LakeEnvironment:
    def __init__(self, params, T=100, seed=None):
        """
        params: dictionary of uncertain parameters
            keys: ['mu', 'sigma', 'b', 'q', 'delta']
        T: simulation horizon
        seed: random seed for reproducibility
        """
        self.mu = params.get('mu', 0.02)
        self.sigma = params.get('sigma', 0.01)
        self.b = params.get('b', 0.42)
        self.q = params.get('q', 2.0)
        self.delta = params.get('delta', 0.98)
        self.T = T
        self.X = np.zeros(T + 1)

        # initialize random inflow generator (correct lognormal mean/var)
        rng = np.random.default_rng(seed)
        self.eps = self._generate_lognormal_inflow(rng)

    def _generate_lognormal_inflow(self, rng):
        """
        Generate lognormal inflows with specified mean (μ)
        and variance (σ²) in the lognormal space.
        """
        mu, sigma = self.mu, self.sigma
        # convert to normal parameters ψ, φ
        phi = np.sqrt(np.log(1 + (sigma**2 / mu**2)))
        psi = np.log(mu) - 0.5 * phi**2
        return rng.lognormal(mean=psi, sigma=phi, size=self.T)
```

```

def X_crit(self):
    """Compute X_crit in the bounded range [0.01, 1.5]."""
    def f(X):
        return (X**self.q) / (1 + X**self.q) - self.b * X

    try:
        Xc = brentq(f, 0.01, 1.5)
    except ValueError:
        # fallback if no root in range
        Xc = 1.5
    return Xc

def update(self, emissions, mitigation, t):
    """Update lake pollution given emissions, mitigation, and random inflow."""
    Xt = self.X[t]
    next_X = Xt + emissions - mitigation + (Xt**self.q) / (1 + Xt**self.q) - self.b * Xt + self.eps[t]
    self.X[t + 1] = max(next_X, 0.0)
    return self.X[t + 1]

# -----
# Stakeholder Base Class
# -----
class Stakeholder:
    def __init__(self, name, delta):
        self.name = name
        self.delta = delta
        self.objectives = {}

    def discount(self, t):
        return self.delta ** t

# -----
# Economic Developer
# -----
class EconomicDeveloper(Stakeholder):
    def __init__(self, alpha_E, delta):
        super().__init__("Economic Developer", delta)
        self.alpha_E = alpha_E

    def effective_emission(self, a_E, u_E):
        return min(a_E, u_E)

    def objective(self, emissions):
        val = sum(self.alpha_E * e * self.discount(t) for t, e in enumerate(emissions))
        self.objectives["economic_benefit"] = val
        return self.objectives

# -----
# Environmental Regulator
# -----
class EnvironmentalRegulator(Stakeholder):
    def __init__(self, alpha_R, delta):
        super().__init__("Environmental Regulator", delta)
        self.alpha_R = alpha_R

    def objectives_fn(self, emissions, X, Xcrit):
        econ_support = sum(self.alpha_R * sum(e) * self.discount(t) for t, e in enumerate(emissions))
        env_protect = -np.mean(X >= Xcrit)
        self.objectives = {
            "economic_support": econ_support,
            "environmental_protection": env_protect,
        }
        return self.objectives

# -----
# Local Community
# -----
class LocalCommunity(Stakeholder):
    def __init__(self, alpha_H, beta_H, delta):

```

```

super().__init__("Local Community", delta)
self.alpha_H = alpha_H
self.beta_H = beta_H

def objectives_fn(self, industrial_emissions, X):
    econ_welfare = sum(self.alpha_H * e * self.discount(t) for t, e in
enumerate(industrial_emissions))
    env_quality = -sum(self.beta_H * x * self.discount(t) for t, x in enumerate(X))
    self.objectives = {
        "economic_wellbeing": econ_welfare,
        "environmental_quality": env_quality,
    }
    return self.objectives

# -----
# Sustainability Advocate (Mitigation / Offset)
# -----
class SustainabilityAdvocate(Stakeholder):
    def __init__(self, gamma_S, delta):
        super().__init__("Sustainability Advocate", delta)
        self.gamma_S = gamma_S

    def effective_mitigation(self, a_S, c_S):
        return min(a_S, c_S)

    def objectives_fn(self, X, mitigation, Xcrit):
        preserve = -np.mean(X >= Xcrit)
        mitigation_cost = -sum(self.gamma_S * m * self.discount(t) for t, m in
enumerate(mitigation))
        self.objectives = {
            "ecological_preservation": preserve,
            "mitigation_cost": mitigation_cost,
        }
        return self.objectives

# -----
# Unified Simulation Interface
# -----
def simulate_lake_system(controls, params, T=100, seed=None):
    """
    Unified interface to simulate the lake problem.

    Inputs:
        controls: dict of control variables
        {
            'a_E': [...],      # emissions by developer
            'a_H': [...],      # emissions by households
            'a_S': [...],      # mitigation by sustainability advocate
            'u_R_E': [...],    # regulator cap (industry)
            'u_R_H': [...],    # regulator cap (households)
            'c_S': [...]       # mitigation capacity
        }

        params: dict of uncertain parameters
        {
            'mu', 'sigma', 'b', 'q', 'delta',
            'alpha_E', 'alpha_R', 'alpha_H', 'beta_H', 'gamma_S'
        }

        T: time horizon
        seed: RNG seed for reproducibility

    Returns:
        dict with objective values for each perspective
    """

    # --- initialize environment ---
    env = LakeEnvironment(params, T=T, seed=seed)
    Xcrit = env.X_crit()

    # --- initialize stakeholders ---
    dev = EconomicDeveloper(params['alpha_E'], env.delta)
    reg = EnvironmentalRegulator(params['alpha_R'], env.delta)

```

```

com = LocalCommunity(params['alpha_H'], params['beta_H'], env.delta)
sus = SustainabilityAdvocate(params['gamma_S'], env.delta)

# --- utility function to vectorize controls ---
def arr(key):
    val = controls.get(key, 0)
    return np.array(val if isinstance(val, (list, np.ndarray)) else [val] * T)

# retrieve control arrays
a_E, a_H, a_S = arr('a_E'), arr('a_H'), arr('a_S')
u_R_E, u_R_H = arr('u_R_E'), arr('u_R_H')
c_S = arr('c_S')

# storage for effective emissions and mitigation
e_E, e_H, m_S = [], [], []

# --- simulate dynamics ---
for t in range(T):
    eE = dev.effective_emission(a_E[t], u_R_E[t])
    eH = min(a_H[t], u_R_H[t])
    mS = sus.effective_mitigation(a_S[t], c_S[t])

    env.update(eE + eH, mS, t)

    e_E.append(eE)
    e_H.append(eH)
    m_S.append(mS)

# --- compute objectives for each stakeholder ---
obj_dev = dev.objective(e_E)
obj_reg = reg.objectives_fn(list(zip(e_E, e_H)), env.X, Xcrit)
obj_com = com.objectives_fn(e_E, env.X)
obj_sus = sus.objectives_fn(env.X, m_S, Xcrit)

# --- compile results ---
return {
    "EconomicDeveloper": obj_dev,
    "EnvironmentalRegulator": obj_reg,
    "LocalCommunity": obj_com,
    "SustainabilityAdvocate": obj_sus,
    "X_crit": Xcrit,
}

# =====
# Example (not executed):
# results = simulate_lake_system(control_dict, param_dict, T=100)
# =====

```

⚙️ Summary of Improvements

Aspect	Before	Now
Threshold estimation	<code>fsolve()</code> (unbounded)	<code>brentq()</code> in [0.01, 1.5]
Lognormal inflow	Interpreted μ , σ as normal parameters	Correctly converts to normal-space parameters (ψ , φ)
Numerical safety	Possible out-of-range or failed roots	Bounded root-finding with fallback
Reproducibility	RNG implicit	Added explicit random seed

✓ This refined version is now **numerically stable**, **physically meaningful**, and **ready for analysis under deep uncertainty**.

It can easily be extended for Monte Carlo runs, policy optimization, or multi-stakeholder coordination analyses.