# Experiments 3 - Market 1

In this problem, four energy producers participate in a day-ahead merit-order electricity market, including three conventional ones and a renewable one. These producers submit hourly bids that specify a delivery quantity of electricity and an offer price for each hour of the next day. The market operator uses these bids to determine the market-clearing price and dispatch schedule. If a bid is accepted, the producer is obligated to deliver the specified quantity of electricity at the clearing price. Otherwise, the producer earns zero profit for that hour.

However, renewable generation is deeply uncertain, as it depends on weather conditions that cannot be predicted. If the producer under-delivers relative to its bid (due to overestimation of production), or over-delivers (due to underestimation), penalties or imbalance costs would apply. These costs are imposed by the system operator to maintain grid stability and to ensure supply-demand balance in real time. Meanwhile, the clearing price is unknown to the producer in advance, so they need to compete in this market.

The renewable energy producer's goal is to maximize its expected profit by strategically selecting the quantity of electricity to bid for each hour of the next day, accounting for the uncertainty in both renewable generation and clearing prices.

Market structure and timing:
1. The day-ahead market is cleared once per day (e.g., at noon) for 24 hourly intervals of the following day.
2. Each producer submits a bid $(b_t, p_t)$ for each hour $t=1, ..., 24$, where:
2.1. $b_t$: the quantity of energy (in MWh) the producer is willing to supply.
2.2. $p_t$: the minimum acceptable price per MWh.
3. The market operator computes the market-clearing price $c_t$ for each hour.
4. If $p_t \leq c_t$, the bid is accepted, and the producer is committed to delivering $b_t$ MWh in hour $t$.
5. The total energy demand in the market is $D_t$, which follows a normal distribution with constant mean $\mu_D$ and variance ${\sigma_D}^2$.

Penalty structure:
1. If actual generation $P_t < b_t$, the shortfall $b_t - P_t$ incurs a penalty of $q_u$ per MWh.
2. If actual generation $P_t \geq b_t$, the surplus $P_t - b_t$ incurs a curtailment penalty of $q_o$ per MWh.

Conventional producers:
1. For each conventional producer $i$, its bid $(b_{it}, p_{it})$ for each hour $t$ follows normal distributions with constant means and variances.

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

Excellent — this is a classic **renewable producer bidding under uncertainty** setup in a day-ahead electricity market.

Let's now carefully **formalize it as a stochastic optimization / decision model**, identifying all **state variables, decisions, uncertainties, and transitions** clearly.

## 🧩 1. Problem setup

We consider a **renewable producer** $R$ participating in a **day-ahead electricity market** for 24 hourly periods $t = 1, \ldots, 24$.

The producer decides bids $(b_t, p_t)$ for each hour before the market clears.
The clearing process and the subsequent realized generation and prices determine the profit.

## 🧮 2. Decision variables

At the start of each day (before market clearing):

$$\begin{aligned} b_t &\in \mathbb{R}_+ \quad &: \text{quantity bid (MWh)} \\ p_t &\in \mathbb{R}_+ \quad &: \text{offer price (\$/MWh)} \end{aligned}$$

Let $\mathbf{b} = (b_1, \ldots, b_{24})$ and $\mathbf{p} = (p_1, \ldots, p_{24})$ be the 24-hour bid vectors.

## 🌤️ 3. Uncertain variables

Each hour $t$ is affected by two key sources of uncertainty:

1. **Renewable generation uncertainty:**

$$P_t \sim F_P(\cdot) \quad \text{(actual renewable output, MWh)}$$

   e.g., normally or beta distributed with mean $\mu_P(t)$ and variance $\sigma_P^2(t)$.

2. **Market price uncertainty:**

$$c_t \sim F_C(\cdot) \quad \text{(market-clearing price, \$/MWh)}$$

   determined endogenously by bids from all producers and random demand $D_t$.

3. **Market demand:**

$$D_t \sim \mathcal{N}(\mu_D, \sigma_D^2)$$

   (affects $c_t$ through the merit-order dispatch).

## ⚙️ 4. Market-clearing mechanism

For each hour $t$:

- The market operator collects all bids $(b_{it}, p_{it})$ from producers $i = 1, \ldots, N$ (including the renewable one).

- The clearing price $c_t$ satisfies:

$$\sum_{i:p_{it} \le c_t} b_{it} = D_t$$

(supply equals demand).

The renewable producer's bid $(b_t, p_t)$ is **accepted** if:

$$p_t \le c_t$$

---

## 💰 5. Profit function (per hour)

Given the realized market price $c_t$ and production $P_t$:

$$\pi_t(b_t, p_t, c_t, P_t) = \begin{cases} c_t b_t - q_u(b_t - P_t), & \text{if } p_t \le c_t,\ P_t < b_t \\ c_t b_t - q_o(P_t - b_t), & \text{if } p_t \le c_t,\ P_t \ge b_t \\ 0, & \text{if } p_t > c_t \end{cases}$$

That is:

- Earns revenue $c_t b_t$ if accepted.
- Pays a **shortfall penalty** $q_u$ per MWh if under-delivery.
- Pays a **curtailment penalty** $q_o$ per MWh if over-delivery.
- Earns nothing if rejected.

---

## 🎯 6. Objective function

The renewable producer's goal is to **maximize expected total daily profit**:

$$\max_{\{b_t, p_t\}_{t=1}^{24}} \mathbb{E}\left[\sum_{t=1}^{24} \pi_t(b_t, p_t, c_t, P_t)\right]$$

subject to:

$$b_t \ge 0, \quad p_t \ge 0, \quad t = 1, \dots, 24$$

Expectation is taken over the joint uncertainty of $(P_t, c_t)$.

---

## 🧠 7. Model components (state-space form)

We can describe this as a **stochastic decision process**:

| Category | Symbol / Definition |
| --- | --- |
| **State variables** | $s_t = (\text{forecast of } P_t, \text{forecast of } c_t, t)$ — captures current knowledge of generation and price distributions at hour $t$. |

| Category | Symbol / Definition |
|---|---|
| **Decision variables** | $a_t = (b_t, p_t)$ — bid quantity and price for hour $t$. |
| **Random variables** | $\omega_t = (P_t, c_t, D_t)$ — realized renewable output, clearing price, and demand. |
| **Transition function** | $s_{t+1} = f(s_t, a_t, \omega_t)$ — updates forecasts for next hour (e.g., based on weather forecast evolution). |
| **Reward function** | $r(s_t, a_t, \omega_t) = \pi_t(b_t, p_t, c_t, P_t)$ |
| **Initial state** | $s_1$ — forecast distributions for renewable generation and prices for hour 1. |
| **Objective** | Maximize $\mathbb{E}\left[\sum_{t=1}^{24} r(s_t, a_t, \omega_t)\right]$. |

---

## 🔁 8. Transition dynamics (optional for sequential formulation)

If hourly decisions are interdependent (e.g., bidding depends on updated forecasts as time advances), we can model transitions as:

$$\begin{aligned}\text{Forecast mean of } P_{t+1} &= \mu_P(t+1 \mid \text{new weather info})\\ \text{Forecast variance of } P_{t+1} &= g(\text{forecast error at } t)\end{aligned}$$

If each hour's bid is decided **independently at noon for all 24 hours**, transitions are trivial (no feedback during the day).

---

## 🧾 9. Compact model summary

$$\begin{aligned}\max_{b_t, p_t} \quad & \sum_{t=1}^{24} \mathbb{E}_{P_t, c_t}[\pi_t(b_t, p_t, c_t, P_t)]\\ \text{s.t.} \quad & b_t, p_t \geq 0,\\ & \pi_t(b_t, p_t, c_t, P_t) = \begin{cases} c_t b_t - q_u(b_t - P_t)^+, & p_t \leq c_t\\ c_t b_t - q_o(P_t - b_t)^+, & p_t \leq c_t\\ 0, & p_t > c_t\end{cases}\end{aligned}$$

where $(x)^+ = \max(x, 0)$.

---

## ✅ 10. Interpretation

- The model captures both **market-clearing uncertainty** (price) and **production uncertainty** (renewables).
- It can be solved via:
  - **Analytical expected profit optimization** (if distributions are known),
  - **Simulation-based optimization**,

- or **stochastic programming / reinforcement learning** depending on information structure.

---

Would you like me to extend this model into a **stochastic program formulation** (e.g., two-stage with scenarios) or into a **dynamic programming (MDP)** structure with sequential bidding decisions?

> There are two areas where you can improve the specification: (1) the renewable energy generation is deeply uncertain, so its probability distribution cannot be determined; (2) for each conventional producer $i$, its bid $(b_{it}, p_{it})$ for each hour $t$ follows normal distributions with constant means and variances. Please refine your specification.

Excellent — you're right to highlight those two subtle but very important modeling refinements:

1. **Renewable generation** is *deeply uncertain* (distribution not well-specified or unreliable).
2. **Conventional producers** have *stochastic bids* with known (normal) distributions.

Let's now **refine the model** accordingly, moving from a purely stochastic optimization framework to a **hybrid model** that explicitly distinguishes **risk/ambiguity** for the renewable and **stochastic but well-characterized uncertainty** for the conventional participants.

---

# 🌍 Refined Model: Renewable Bidding under Deep and Stochastic Uncertainty

We consider a **renewable producer** $R$ and three **conventional producers** $i = 1, 2, 3$ participating in a **day-ahead electricity market** for $t = 1, \dots, 24$.

---

## 1. Decision Variables

At the day-ahead bidding stage (before clearing):

$$\begin{aligned} b_t &\geq 0 \quad : \text{quantity bid by the renewable producer (MWh)} \\ p_t &\geq 0 \quad : \text{offer price per MWh} \end{aligned}$$

Decision vector: $a_t = (b_t, p_t)$.

---

## 2. Uncertain Elements

### (a) Market demand uncertainty (stochastic, well-characterized):

$$D_t \sim \mathcal{N}(\mu_D, \sigma_D^2)$$

### (b) Conventional producers' bids (stochastic, known distributions):

For each conventional producer $i = 1, 2, 3$:

$$b_{it} \sim \mathcal{N}(\mu_{b_i}, \sigma_{b_i}^2), \quad p_{it} \sim \mathcal{N}(\mu_{p_i}, \sigma_{p_i}^2)$$

and assumed *independent across producers and hours*.

**(c) Renewable generation uncertainty (deep uncertainty):**

- Actual renewable generation $P_t$ is *not drawn from a known probability distribution*.
- Instead, we model it as an *ambiguous quantity* whose realization lies within an **uncertainty set**:

  $$P_t \in \mathcal{U}_t = [\underline{P}_t, \overline{P}_t]$$

  or more generally, as belonging to a *family of plausible distributions* $\mathcal{F}_t$.

Thus, we cannot assign precise probabilities — the renewable must be robust against possible realizations.

---

## 3. Market Clearing Process

For each hour $t$:

The market operator sets a clearing price $c_t$ such that:

$$\sum_{i=1}^{3} b_{it}\,\mathbf{1}(p_{it} \le c_t) + b_t\,\mathbf{1}(p_t \le c_t) = D_t$$

Given the conventional producers' stochastic bids, $c_t$ is a **random variable** with a distribution induced by $(b_{it}, p_{it}, D_t)$.

We can write:

$$c_t = h_t(D_t, \{b_{it}, p_{it}\}_{i=1}^3, b_t, p_t)$$

where $h_t(\cdot)$ is the **market-clearing function**.

---

## 4. Payoff and Penalties

Given realized generation $P_t$, clearing price $c_t$, and bid $(b_t, p_t)$:

$$\pi_t(b_t, p_t, c_t, P_t) = \begin{cases} c_t b_t - q_u(b_t - P_t), & \text{if } p_t \le c_t,\ P_t < b_t \\ c_t b_t - q_o(P_t - b_t), & \text{if } p_t \le c_t,\ P_t \ge b_t \\ 0, & \text{if } p_t > c_t \end{cases}$$

or equivalently:

$$\pi_t = \mathbf{1}(p_t \le c_t)\left[c_t b_t - q_u(b_t - P_t)^+ - q_o(P_t - b_t)^+\right]$$

---

## 5. Objective Function: Robust–Stochastic Optimization

The renewable producer faces:

- **Stochastic uncertainty** from market prices $c_t$ (driven by conventional bids and demand),
- **Deep uncertainty** from own production $P_t$.

Hence, the **expected profit** is taken over *known stochastic elements*, while being *robust* to the deep uncertainty in $P_t$:

$$\max_{\{b_t,p_t\}} \mathbb{E}_{c_t}\left[\min_{P_t\in\mathcal{U}_t} \pi_t(b_t,p_t,c_t,P_t)\right]$$

for each hour $t$, or jointly over all hours:

$$\max_{\{b_t,p_t\}_{t=1}^{24}} \sum_{t=1}^{24} \mathbb{E}_{c_t}\left[\min_{P_t\in\mathcal{U}_t} \pi_t(b_t,p_t,c_t,P_t)\right]$$

If the renewable producer considers ambiguity across possible distributions instead of intervals, we may write a **distributionally robust** version:

$$\max_{\{b_t,p_t\}_{t=1}^{24}} \sum_{t=1}^{24} \min_{F_P\in\mathcal{F}_t} \mathbb{E}_{c_t,P_t\sim F_P}[\pi_t(b_t,p_t,c_t,P_t)]$$

## 6. State-Space Representation

| Category | Symbol / Definition |
|---|---|
| **State variables** | $s_t = (\text{forecast range of } P_t, \text{distribution of } c_t, t)$ |
| **Decision variables** | $a_t = (b_t, p_t)$ |
| **Random variables (stochastic)** | $\omega_t^S = (D_t, \{b_{it}, p_{it}\}_{i=1}^3)$ |
| **Uncertain parameters (deep)** | $\omega_t^D = P_t \in \mathcal{U}_t \text{ or } F_P \in \mathcal{F}_t$ |
| **Transition function** | $s_{t+1} = f(s_t, a_t, \omega_t^S, \omega_t^D)$ (update forecasts, uncertainty sets) |
| **Reward function** | $r(s_t, a_t, \omega_t^S, \omega_t^D) = \pi_t(b_t, p_t, c_t, P_t)$ |
| **Initial state** | $s_1 = (\text{initial forecast bounds on } P_1, \text{price distribution}, t = 1)$ |
| **Objective** | $\max \mathbb{E}_{\omega^S} \min_{\omega^D\in\mathcal{U}} \sum_t r(s_t, a_t, \omega_t^S, \omega_t^D)$ |

## 7. Interpretation

- **Price side:** conventional producers' normal-distributed bids make the clearing price stochastic but analyzable.
- **Production side:** renewable generation is *deeply uncertain*, requiring **robust or distributionally robust bidding**.
- **Decision logic:** the renewable producer maximizes expected profit *against known market stochasticity* while *hedging* against *worst-case feasible generation outcomes*.

## 8. Compact Formulation

$$
\begin{aligned}
\max_{\{b_t,p_t\}} \quad & \sum_{t=1}^{24} \mathbb{E}_{c_t\sim F_C}\left[\min_{P_t\in\mathcal{U}_t} \mathbf{1}(p_t \le c_t)\left(c_t b_t - q_u(b_t - P_t)^+ - q_o(P_t - b_t)^+\right)\right] \\
\text{s.t.} \quad & \begin{cases} b_t, p_t \ge 0, & t = 1,\dots,24, \\ c_t = h_t(D_t, \{b_{it}, p_{it}\}_{i=1}^3, b_t, p_t), \\ D_t \sim \mathcal{N}(\mu_D, \sigma_D^2),\ (b_{it}, p_{it}) \sim \mathcal{N}(\mu_{b_i}, \sigma_{b_i}^2) \times \mathcal{N}(\mu_{p_i}, \sigma_{p_i}^2). \end{cases}
\end{aligned}
$$

Would you like me to extend this further into a **two-stage distributionally robust stochastic program** (with scenario-based market simulation for the stochastic part and uncertainty sets for $P_t$)?
That would make it directly ready for computational formulation.

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

Excellent — now we're elevating the problem from a single-agent decision under uncertainty into a **multi-stakeholder dynamic system**.

We will now formalize a **two-agent, shared-environment model**:

- **Agent 1:** the **Renewable Producer (RP)** — a market participant optimizing its bidding strategy under uncertainty.
- **Agent 2:** the **System Regulator (SR)** — representing the market operator or regulator who designs rules, penalties, or incentives to ensure system reliability, efficiency, and fairness.

Both agents **act in the same environment** (the electricity market), sharing key global parameters and variables such as total demand, market prices, and renewable output uncertainty.
They **do not control each other directly**, but each one's decision affects the other through these shared market mechanisms.

## ⚙️ 1. Common Environment (Shared System)

Let the environment $\mathcal{E}$ capture all exogenous processes and global constraints shared by both stakeholders.

### 1.1 Global parameters

| Symbol | Description |
|---|---|
| $T = \{1, \dots, 24\}$ | Set of hourly time periods |
| $N_c = 3$ | Number of conventional producers |
| $\mu_D, \sigma_D^2$ | Mean and variance of demand $D_t \sim \mathcal{N}(\mu_D, \sigma_D^2)$ |
| $q_u, q_o$ | Penalty rates for under-delivery and over-delivery (MWh) |
| $\mu_{b_i}, \sigma_{b_i}^2, \mu_{p_i}, \sigma_{p_i}^2$ | Mean and variance parameters of conventional producers' bids |
| $\mathcal{U}_t$ | Uncertainty set (range or ambiguity set) for renewable generation $P_t$ |

### 1.2 Global variables

| Variable | Description |
|---|---|
| $D_t$ | Total system demand (stochastic) |
| $b_{it}, p_{it}$ | Conventional producers' bids (stochastic) |
| $b_t, p_t$ | Renewable producer's bid (decision variable for RP) |
| $P_t$ | Realized renewable generation (deeply uncertain) |
| $c_t$ | Market-clearing price (determined endogenously) |
| $A_t \subseteq \{\text{accepted producers}\}$ | Set of accepted bids at hour $t$ |
| $\Pi_t^R, \Pi_t^C$ | Profits of renewable and conventional producers |
| $\mathcal{S}_t$ | System state at hour $t$ (encapsulating all relevant global information) |

### 1.3 Market clearing rule (shared mechanism)

The **market-clearing price** $c_t$ satisfies:

$$\sum_{i=1}^{N_c} b_{it}\mathbf{1}(p_{it} \le c_t) + b_t\mathbf{1}(p_t \le c_t) = D_t$$

All accepted bids must jointly meet the total system demand.

This **market equilibrium** acts as the **link** between the renewable producer's and the regulator's models.

## 🌞 2. Renewable Producer's Model (Agent 1)

### 2.1 Control variables

$$a_t^R = (b_t, p_t)$$

- Quantity bid $b_t$
- Offer price $p_t$

### 2.2 Uncertain inputs

$$\omega_t^R = (c_t, P_t)$$

where $c_t$ is stochastic (market price from environment) and $P_t$ is deeply uncertain (generation).

### 2.3 State variables

$$s_t^R = (\text{forecast of renewable potential at } t, \text{forecast of price distribution}, t)$$

### 2.4 Transition function

$$s_{t+1}^R = f^R(s_t^R, a_t^R, \omega_t^R)$$

Forecasts evolve as time progresses (e.g., weather forecast updates).

### 2.5 Objective(s)

The renewable producer seeks to **maximize profitability** under market and generation uncertainty. Because generation is deeply uncertain, this involves **robust or distributionally robust optimization**.

**Objective 1 — Expected profit (over market stochasticity):**

$$J_1^R = \mathbb{E}_{c_t}\left[\mathbf{1}(p_t \leq c_t)\left(c_t b_t - q_u(b_t - P_t)^+ - q_o(P_t - b_t)^+\right)\right]$$

**Objective 2 — Robustness against generation uncertainty:**

$$J_2^R = \min_{P_t \in \mathcal{U}_t} J_1^R$$

**Decision problem (per day):**

$$\max_{\{b_t, p_t\}_{t=1}^{24}} \sum_{t=1}^{24} J_2^R$$

or equivalently:

$$\boxed{\max_{b_t, p_t} \mathbb{E}_{c_t}\left[\min_{P_t \in \mathcal{U}_t} \pi_t(b_t, p_t, c_t, P_t)\right]}$$

---

## ⚖️ 3. System Regulator's Model (Agent 2)

The **system regulator** represents the entity responsible for ensuring **market efficiency, reliability, and fairness**, given that producers act strategically.

### 3.1 Control variables

$$a_t^S = (q_u, q_o, \tau_t, \rho_t)$$

where:

- $q_u, q_o$: penalty coefficients (controllable policy levers),
- $\tau_t$: renewable curtailment threshold or system reliability constraint (e.g., acceptable imbalance),

- $\rho_t$: market rule parameter (e.g., reserve requirement or price cap).

### 3.2 State variables

$$s_t^S = (\text{system reliability metrics, aggregate supply/demand}, t)$$

Example components:

- Expected imbalance $I_t = |\sum_{i \in A_t} b_{it} - D_t|$
- Reserve margin $R_t$

### 3.3 Transition function

$$s_{t+1}^S = f^S(s_t^S, a_t^S, \{a_t^R, a_{it}^C\}, \omega_t^S)$$

- Captures how system state evolves based on regulator's and producers' decisions.
- For example, changes in penalties influence future bidding behavior and imbalance levels.

### 3.4 Objective(s)

The regulator typically has *multiple competing objectives*.

**Objective 1 — Reliability (minimize imbalance)**

$$J_1^S = \mathbb{E}[I_t^2] = \mathbb{E}\left[\left(D_t - \sum_{i \in A_t} b_{it}\right)^2\right]$$

**Objective 2 — Market efficiency (minimize total cost)**

$$J_2^S = \mathbb{E}\left[\sum_{i \in A_t} c_t b_{it}\right]$$

**Objective 3 — Fairness / stability**

$$J_3^S = \mathrm{Var}(c_t) \quad (\text{reduce price volatility})$$

**Objective 4 — Renewable integration**

$$J_4^S = -\mathbb{E}[b_t \mathbf{1}(p_t \leq c_t)] \quad (\text{maximize accepted renewable share})$$

Each objective can be addressed separately (no weighted sum).

The regulator's **decision problem** is then:

$$\boxed{\min_{a_t^S} \{J_1^S, J_2^S, J_3^S, J_4^S\}}$$

subject to the market-clearing condition and producers' bidding responses.

# 🔗 4. Coupling Between Stakeholders

The **renewable producer** and the **regulator** are coupled *only through shared environmental variables*:

| Shared variable | Determined by | Influences |
|---|---|---|
| Market-clearing price $c_t$ | Environment (via bids and demand) | RP's profit, SR's cost |
| Penalties $q_u, q_o$ | Regulator | RP's bid decision |
| Accepted quantities $b_t \mathbf{1}(p_t \leq c_t)$ | Market operator | System reliability metrics |
| Realized generation $P_t$ | Nature (deep uncertainty) | RP's profit, SR's imbalance |
| Demand $D_t$ | Environment | Both indirectly via price clearing |

Thus:

- The **Renewable Producer** acts as a **price-taker** but reacts strategically to penalty policies.
- The **System Regulator** acts as a **policy-maker**, shaping $q_u, q_o, \rho_t$ to influence producer behavior and system outcomes.

---

# 🧩 5. Summary: Two-Agent Shared Environment Model

| Component | Renewable Producer (RP) | System Regulator (SR) | Shared Environment |
|---|---|---|---|
| **State** | $s_t^R = (\text{price forecast, generation range}, t)$ | $s_t^S = (\text{imbalance, cost}, t)$ | $\mathcal{S}_t = (D_t, c_t, P_t, \{b_{it}, p_{it}\})$ |
| **Control variables** | $a_t^R = (b_t, p_t)$ | $a_t^S = (q_u, q_o, \tau_t, \rho_t)$ | — |
| **Transition** | $s_{t+1}^R = f^R(s_t^R, a_t^R, \mathcal{S}_t)$ | $s_{t+1}^S = f^S(s_t^S, a_t^S, \mathcal{S}_t)$ | $\mathcal{S}_{t+1} = g(\mathcal{S}_t, a_t^R, a_t^S)$ |
| **Objectives** | $J_1^R$: profit expectation $J_2^R$: robustness | $J_1^S$: reliability $J_2^S$: efficiency $J_3^S$: stability $J_4^S$: renewable integration | Shared through $c_t$, $D_t$, $P_t$, $q_u, q_o$ |
| **Interaction** | Reacts to market price & penalties | Sets penalties and rules | Market clearing links them |

---

# 🧠 Conceptual Summary

This specification defines a **bi-level, multi-objective stochastic-robust system**:

- **Lower level (renewable producer):**
  Robust-stochastic optimization under uncertain prices and deeply uncertain production.
- **Upper level (regulator):**
  Multi-objective control problem shaping penalty and market design to minimize imbalances and ensure efficient, fair market outcomes.

Both operate within the **shared environment** of the day-ahead electricity market, connected through endogenous prices and uncertainty propagation.

---

Would you like me to extend this into a **bi-level mathematical program** (explicitly expressing the renewable producer's optimization as the lower level of the regulator's problem)? That would formalize it as a regulatory Stackelberg game.

Your specification should only include implementable and quantifiable components. For those conceptual components, please refine them or remove them.

Excellent — that's a crucial refinement. Let's now reformulate the model to **retain only implementable, measurable, and quantifiable components**, removing conceptual or purely qualitative notions (like "fairness" or "stability") and replacing them with quantifiable operational metrics.

We will keep:

- Clear **mathematical structure** for both the **renewable producer** and the **system regulator**,
- A **shared, measurable environment**,
- Well-defined **control variables**, **state transitions**, and **objective functions** that are **directly computable from observable data**.

---

## ⚙️ 1. Common Environment (Shared System)

Both stakeholders operate in a **shared stochastic-robust market environment**

$$\mathcal{E} = \{D_t, P_t, b_{it}, p_{it}, c_t, q_u, q_o\}_{t=1}^{24}$$

### 1.1 Global Parameters (fixed and observable)

| Symbol | Description | Units / Type |
|---|---|---|
| $T = \{1, \dots, 24\}$ | Set of hourly intervals | — |
| $N_c = 3$ | Number of conventional producers | integer |
| $\mu_D, \sigma_D^2$ | Mean and variance of demand $D_t$ | MWh, MWh² |
| $\mu_{b_i}, \sigma_{b_i}^2$ | Mean and variance of bid quantities for conventional producer $i$ | MWh, MWh² |
| $\mu_{p_i}, \sigma_{p_i}^2$ | Mean and variance of bid prices for producer $i$ | \$/MWh, (\$/MWh)² |
| $q_u, q_o$ | Penalty rates for under- and over-delivery (controlled by regulator) | \$/MWh |
| $\underline{P}_t, \overline{P}_t$ | Lower/upper bounds of renewable generation uncertainty set | MWh |

---

### 1.2 Global Random Variables (stochastic components)

| Symbol | Description | Distribution |
|---|---|---|
| $D_t$ | Market demand | $\mathcal{N}(\mu_D, \sigma_D^2)$ |
| $b_{it}, p_{it}$ | Conventional producers' bids | $b_{it} \sim \mathcal{N}(\mu_{b_i}, \sigma_{b_i}^2)$, $p_{it} \sim \mathcal{N}(\mu_{p_i}, \sigma_{p_i}^2)$ |

---

OpenAI

### 1.3 Market-Clearing Mechanism (shared function)

At each hour $t$, the market operator computes the clearing price $c_t$ from bids and demand:

$$\sum_{i=1}^{N_c} b_{it}\mathbf{1}(p_{it} \le c_t) + b_t\mathbf{1}(p_t \le c_t) = D_t$$

This determines:

- Market-clearing price $c_t$,
- Which producers' bids are accepted.

This function $c_t = h_t(D_t, \{b_{it}, p_{it}\}, b_t, p_t)$ is **fully computable** from realizations of bids and demand.

---

### 1.4 Realized Renewable Generation (deep uncertainty)

$$P_t \in [\underline{P}_t, \overline{P}_t]$$

The renewable producer only knows the feasible range but not the probability distribution of $P_t$.

---

## 🌞 2. Renewable Producer (Agent 1)

### 2.1 Control Variables

$$a_t^R = (b_t, p_t)$$

- $b_t$: bid quantity (MWh)
- $p_t$: offer price ($/MWh)

### 2.2 Observed Inputs

- Forecasted clearing price distribution $F_{c_t}$ from market data.
- Forecasted renewable potential range $[\underline{P}_t, \overline{P}_t]$.

### 2.3 State Variables

$$s_t^R = (\underline{P}_t, \overline{P}_t, F_{c_t}, t)$$

All quantities measurable or inferable before bidding.

### 2.4 Transition Function

If bidding occurs once per day, no dynamic transition within the day:

$$s_{t+1}^R = s_t^R \quad \forall t$$

If forecasts update hourly (optional extension):

$$s_{t+1}^R = f^R(s_t^R) = (\underline{P}_{t+1}, \overline{P}_{t+1}, F_{c_{t+1}}, t+1)$$

### 2.5 Profit Function (per hour)

$$\pi_t(b_t, p_t, c_t, P_t) = \mathbf{1}(p_t \le c_t)\left[c_t b_t - q_u(b_t - P_t)^+ - q_o(P_t - b_t)^+\right]$$

### 2.6 Objective Function(s)

Since $P_t$ is uncertain but bounded, the renewable producer's objective is **robust expected profit maximization**:

**(1) Expected profit over stochastic prices:**

$$J_1^R = \mathbb{E}_{c_t}[\pi_t(b_t, p_t, c_t, P_t)]$$

**(2) Robust profit over uncertain generation:**

$$J_2^R = \min_{P_t \in [\underline{P}_t, \overline{P}_t]} J_1^R$$

**Overall daily problem:**

$$\boxed{\max_{\{b_t, p_t\}_{t=1}^{24}} \sum_{t=1}^{24} \mathbb{E}_{c_t}\left[\min_{P_t \in [\underline{P}_t, \overline{P}_t]} \pi_t(b_t, p_t, c_t, P_t)\right]}$$

All terms are measurable: bids, penalties, clearing price, and power range.

---

## ⚖️ 3. System Regulator (Agent 2)

### 3.1 Control Variables

$$a_t^S = (q_u, q_o)$$

- $q_u$: penalty for under-delivery ($/MWh)
- $q_o$: penalty for over-delivery ($/MWh)

These can be adjusted across days or hours.

### 3.2 State Variables

$$s_t^S = (D_t, c_t, \text{imbalance}_t, t)$$

where

$$\text{imbalance}_t = D_t - \sum_{i=1}^{N_c} b_{it}\mathbf{1}(p_{it} \le c_t) - b_t\mathbf{1}(p_t \le c_t)$$

—all observable after market clearing.

### 3.3 Transition Function

The system state updates deterministically from the previous hour's cleared results:

$$s_{t+1}^S = f^S(s_t^S, a_t^S) = (D_{t+1}, c_{t+1}, \text{imbalance}_{t+1}, t+1)$$

where $D_{t+1}$ and bids are new realizations drawn from known distributions.

### 3.4 Objective Function(s)

All regulator objectives are quantifiable using observed variables.

**(1) Reliability Objective – Minimize expected imbalance magnitude**

$$J_1^S = \mathbb{E}[|\text{imbalance}_t|]$$

**(2) Cost Efficiency – Minimize total expected procurement cost**

$$J_2^S = \mathbb{E}[c_t D_t]$$

**(3) Market Participation – Maximize renewable acceptance**

$$J_3^S = -\mathbb{E}[b_t \mathbf{1}(p_t \le c_t)]$$

(Equivalent to minimizing negative renewable share.)

The regulator's optimization problem is thus:

$$\boxed{\min_{q_u, q_o} \ \{J_1^S, J_2^S, J_3^S\}}$$

with each objective assessed empirically via market outcomes.

---

## 🔗 4. Interaction via Shared Environment

| Shared Variable | Determined by | Affects |
|---|---|---|
| $c_t$ (clearing price) | Market-clearing mechanism using all bids | RP profit, SR cost |
| $q_u, q_o$ (penalties) | Regulator control | RP bidding behavior |
| $P_t$ (renewable output) | Exogenous environment | RP profit, SR imbalance |
| $D_t$ (demand) | Random | Price clearing, imbalance |
| Accepted bids | Both (through $p_t, q_u, q_o$) | Total system cost, reliability |

The system is a **quantifiable two-agent game** with endogenous coupling through $c_t$, $q_u, q_o$, and market equilibrium.

---

## 🧩 5. Combined Implementable Framework Summary

| Component | Renewable Producer (RP) | System Regulator (SR) | Shared Environment |
|---|---|---|---|
| **Control Variables** | $b_t, p_t$ | $q_u, q_o$ | — |

| Component | Renewable Producer (RP) | System Regulator (SR) | Shared Environment |
|---|---|---|---|
| **State Variables** | $[\underline{P}_t, \overline{P}_t], F_{c_t}$ | $D_t, c_t, \text{imbalance}_t$ | $D_t, P_t, \{b_{it}, p_{it}\}$ |
| **Transition Function** | Forecast updates $f^R$ | Market evolution $f^S$ | Price clearing $h_t$ |
| **Objective Functions** | $J_1^R$: expected profit; $J_2^R$: robust profit | $J_1^S$: imbalance; $J_2^S$: cost; $J_3^S$: renewable share | — |
| **Coupling** | Via $c_t, q_u, q_o$ | Via $b_t, p_t$ | Price clearing, penalties |

---

## ✅ This refined specification is:

- **Quantifiable:** all terms (bids, prices, penalties, imbalance, costs) are measurable from data.
- **Implementable:** all control and state variables can be observed or estimated operationally.
- **Structured:** captures two independent decision makers interacting through a shared stochastic-robust environment.

---

Would you like me to now write this as a **bi-level mathematical program (Stackelberg game)** using these quantifiable definitions — with the regulator at the upper level and the renewable producer at the lower level?

> 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 the right next step.
We'll now **compose a unified, internally consistent model** that embeds:

- both **stakeholder perspectives** (Renewable Producer and System Regulator),
- a **shared environment and state**,
- perspective-specific **control variables** and **objectives**,
- and a structure allowing **modular evaluation** — each stakeholder treats the other's controls as exogenous when optimizing.

---

# 🧩 Unified Shared-Environment Market Model

We denote the overall system as a **shared stochastic–robust environment** with two agents:

- **Agent R** – Renewable Producer (RP)

- **Agent S** – System Regulator (SR)

They act on a **common state space**, sharing all environmental variables, but each controls its own variables and optimizes its own objective(s).

The resulting model is both **modular** (each perspective analyzable independently) and **internally coupled** through shared variables (prices, penalties, demand).

---

## 1️⃣ Shared Environment $\mathcal{E}$

The environment represents the **day-ahead electricity market** for $t = 1, \ldots, 24$ hours.

### 1.1 Global Parameters (constants)

| Symbol | Description | Units |
|---|---|---|
| $T = \{1, \ldots, 24\}$ | Time horizon (hours) | — |
| $N_c = 3$ | Number of conventional producers | — |
| $\mu_D, \sigma_D^2$ | Mean/variance of demand | MWh, MWh² |
| $\mu_{b_i}, \sigma_{b_i}^2$ | Mean/variance of bid quantity for producer i | MWh, MWh² |
| $\mu_{p_i}, \sigma_{p_i}^2$ | Mean/variance of bid price for producer i | $/MWh, ($/MWh)² |
| $[\underline{P}_t, \overline{P}_t]$ | Renewable generation range (deep uncertainty) | MWh |

---

### 1.2 Global Random Variables (stochastic processes)

| Symbol | Description | Distribution |
|---|---|---|
| $D_t$ | Total system demand | $\mathcal{N}(\mu_D, \sigma_D^2)$ |
| $b_{it}, p_{it}$ | Conventional producers' bids | $b_{it} \sim \mathcal{N}(\mu_{b_i}, \sigma_{b_i}^2)$, $p_{it} \sim \mathcal{N}(\mu_{p_i}, \sigma_{p_i}^2)$ |

---

### 1.3 Shared State Variables $s_t$

At any hour $t$, the **common state** includes all observable quantities:

$$s_t = (t, D_t, \{b_{it}, p_{it}\}_{i=1}^{N_c}, [\underline{P}_t, \overline{P}_t], c_t, q_u, q_o)$$

- $c_t$: market-clearing price (endogenous variable),
- $q_u, q_o$: penalty rates (regulator's controls, shared through the environment).

---

### 1.4 Market-Clearing Function (shared transition rule)

The market operator computes $c_t$ such that total accepted supply meets demand:

$$\sum_{i=1}^{N_c} b_{it} \mathbf{1}(p_{it} \leq c_t) + b_t \mathbf{1}(p_t \leq c_t) = D_t$$

The resulting price $c_t = h_t(D_t, \{b_{it}, p_{it}\}, b_t, p_t)$ defines the **state transition** for the shared environment.

---

### 1.5 State Transition Function for the Environment

For each period:

$$s_{t+1} = g(s_t, a_t^R, a_t^S, \omega_{t+1})$$

where $a_t^R, a_t^S$ are the agents' controls,
and $\omega_{t+1} = (D_{t+1}, \{b_{i,t+1}, p_{i,t+1}\})$ are new stochastic inputs.

$g(\cdot)$ updates:

- demand and bid realizations,
- recomputed clearing price,
- any regulator parameters if adaptive.

---

## 2️⃣ Renewable Producer Sub-Model (Agent R)

### 2.1 Control Variables

$$a_t^R = (b_t, p_t)$$

Chosen by the renewable producer before market clearing.

### 2.2 Inputs from Shared Environment

$$(D_t, \{b_{it}, p_{it}\}, q_u, q_o, [\underline{P}_t, \overline{P}_t])$$

The producer treats $q_u, q_o$ as **exogenous** parameters set by the regulator.

### 2.3 Profit Function (per hour)

$$\pi_t^R(b_t, p_t, c_t, P_t) = \mathbf{1}(p_t \le c_t)\left[c_t b_t - q_u(b_t - P_t)^+ - q_o(P_t - b_t)^+\right]$$

### 2.4 Objectives

Because renewable output $P_t$ is uncertain but bounded, the producer uses a **robust–stochastic criterion**:

$$J_1^R = \mathbb{E}_{c_t}[\pi_t^R(b_t, p_t, c_t, P_t)] \quad \text{(expected profit)}$$

$$J_2^R = \min_{P_t \in [\underline{P}_t, \overline{P}_t]} J_1^R \quad \text{(worst-case profit)}$$

$$\text{Daily optimization:} \quad \max_{\{b_t, p_t\}_{t=1}^{24}} \sum_{t=1}^{24} J_2^R$$

**Quantifiable outputs:** bid schedule $(b_t, p_t)$, realized profit, expected profit, and worst-case profit.

# 3️⃣ System Regulator Sub-Model (Agent S)

## 3.1 Control Variables

$$a_t^S = (q_u, q_o)$$

Penalty parameters chosen before market clearing (may be hourly or daily constants).

## 3.2 Inputs from Shared Environment

$$(b_t, p_t, D_t, \{b_{it}, p_{it}\})$$

The regulator treats producer bids $(b_t, p_t)$ as **exogenous** when evaluating system performance.

## 3.3 System-Level Quantities (derived from shared state)

$$\begin{aligned}
\text{imbalance}_t &= D_t - \sum_{i=1}^{N_c} b_{it}\mathbf{1}(p_{it} \le c_t) - b_t\mathbf{1}(p_t \le c_t) \\
\text{system cost}_t &= c_t D_t \\
\text{accepted renewables}_t &= b_t\mathbf{1}(p_t \le c_t)
\end{aligned}$$

All measurable after clearing.

## 3.4 Objectives (quantifiable)

$$\begin{aligned}
J_1^S &= \mathbb{E}[|\text{imbalance}_t|] && \text{(reliability)} \\
J_2^S &= \mathbb{E}[\text{system cost}_t] && \text{(economic efficiency)} \\
J_3^S &= -\mathbb{E}[\text{accepted renewables}_t] && \text{(renewable integration; minimize the negative)}
\end{aligned}$$

## 3.5 Regulator's Problem

$$\boxed{\min_{q_u, q_o} \{J_1^S, J_2^S, J_3^S\}}$$

Each objective can be evaluated or optimized separately using observed system data.

# 4️⃣ Composed Unified Model

The unified system couples both perspectives within the same environment.

## 4.1 Joint State and Controls

$$\begin{aligned}
s_t &= (t, D_t, \{b_{it}, p_{it}\}, [\underline{P}_t, \overline{P}_t], c_t, q_u, q_o) \\
a_t &= (a_t^R, a_t^S) = (b_t, p_t, q_u, q_o)
\end{aligned}$$

## 4.2 Shared Transition Function

$$s_{t+1} = g(s_t, a_t, \omega_{t+1})$$

where $g$ computes new demand, bids, and clearing price via:

$$c_t = h_t(D_t, \{b_{it}, p_{it}\}, b_t, p_t)$$

and updates regulator parameters if they vary.

### 4.3 Perspective-Specific Objectives

$$\begin{aligned}
&\textbf{Renewable Producer:} \quad && \max_{\{b_t,p_t\}} \sum_t \mathbb{E}_{c_t}\left[\min_{P_t \in [\underline{P}_t, \overline{P}_t]} \pi_t^R(b_t, p_t, c_t, P_t)\right] \\
&\textbf{System Regulator:} \quad && \min_{q_u,q_o} \{\mathbb{E}[|\text{imbalance}_t|],\ \mathbb{E}[c_t D_t],\ -\mathbb{E}[b_t \mathbf{1}(p_t \le c_t)]\}
\end{aligned}$$

### 4.4 Interaction Rules

- $c_t$ depends on both agents' controls through market clearing.
- $q_u, q_o$ enter the renewable producer's profit function.
- Both agents observe the same state $s_t$.
- Each agent's optimization treats the other's controls as **exogenous inputs**.

---

## 5️⃣ Modular Evaluation

Each module can be simulated or optimized independently:

| Module | Inputs | Outputs | Objective(s) |
|---|---|---|---|
| Renewable Producer | $s_t, q_u, q_o$ | $(b_t, p_t)$, $J_1^R, J_2^R$ | Profit, Robust Profit |
| System Regulator | $s_t, b_t, p_t$ | $q_u, q_o$, $J_1^S, J_2^S, J_3^S$ | Reliability, Cost, Integration |
| Shared Environment | Random draws of $D_t, b_{it}, p_{it}$, generation range | Updated state $s_{t+1}$, $c_t$ | — |

Each stakeholder's model can thus be **evaluated or solved separately**, then combined iteratively (e.g., in simulation or bilevel optimization).

---

### ✅ Unified Model Summary

$$
\boxed{\begin{aligned}
&\text{State: } s_t = (t, D_t, \{b_{it}, p_{it}\}, [\underline{P}_t, \overline{P}_t], c_t, q_u, q_o) \\
&\text{Transition: } s_{t+1} = g(s_t, a_t, \omega_{t+1}) \\
&\text{Controls: } \begin{cases} a_t^R = (b_t, p_t) & \text{(Renewable Producer)} \\ a_t^S = (q_u, q_o) & \text{(System Regulator)} \end{cases} \\
&\text{Objectives:} \\
&\quad J_R = \sum_t \mathbb{E}_{c_t}\left[ \min_{P_t \in [\underline{P}_t, \overline{P}_t]} \pi_t^R(b_t, p_t, c_t, P_t) \right] \\
&\quad J_S = \{\, \mathbb{E}[|\text{imbalance}_t|],\ \mathbb{E}[c_t D_t],\ -\mathbb{E}[b_t \mathbf{1}(p_t \le c_t)] \,\}
\end{aligned}}
$$

This unified formulation:

1. **Embeds** both stakeholders' controls and objectives.
2. **Shares** a quantifiable, consistent environment and state.
3. **Allows modular evaluation** by treating other agents' controls as exogenous.
4. **Includes all measurable components** needed for simulation, calibration, and optimization in real-world settings.

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

Here's a **modular, fully quantifiable Python implementation** of the **unified shared-environment model**, designed for simulation, experimentation, or optimization.

Each stakeholder — the **Renewable Producer** and the **System Regulator** — is implemented as a separate class with its own control variables, objectives, and interfaces.
A **shared `MarketEnvironment`** coordinates interactions and market clearing.
Finally, a **single interface function** allows external evaluation of the model.

---

```python
import numpy as np

# ============================================================
# 1. SHARED MARKET ENVIRONMENT
# ============================================================

class MarketEnvironment:
    """
    Shared environment representing the day-ahead electricity market.
    Stores stochastic demand, conventional bids, renewable uncertainty,
    and market-clearing logic.
    """
    def __init__(self, params, uncertainties):
        """
```

```
        params: dict of fixed parameters (means, variances, time horizon, etc.)
        uncertainties: dict of uncertain variables (e.g., realized demand, conventional bids)
        """
        self.params = params
        self.uncertainties = uncertainties
        self.state = {}

    def sample_demand(self, t):
        mu_D, sigma_D = self.params["mu_D"], self.params["sigma_D"]
        return np.random.normal(mu_D, sigma_D)

    def sample_conventional_bids(self):
        """Generate random bids from normal distributions for conventional producers."""
        N_c = self.params["N_c"]
        bids = []
        for i in range(N_c):
            b_i = np.random.normal(self.params["mu_b"][i], self.params["sigma_b"][i])
            p_i = np.random.normal(self.params["mu_p"][i], self.params["sigma_p"][i])
            bids.append((b_i, p_i))
        return bids

    def market_clearing_price(self, D_t, conv_bids, b_t, p_t):
        """
        Solve for clearing price c_t by sorting offers until cumulative supply meets demand.
        """
        offers = conv_bids + [(b_t, p_t)]
        # sort offers by price
        offers.sort(key=lambda x: x[1])
        supply = 0
        for b_i, p_i in offers:
            supply += b_i
            if supply >= D_t:
                return p_i  # market-clearing price
        return offers[-1][1]  # fallback: highest offer price

    def update_state(self, t, D_t, conv_bids, c_t, q_u, q_o):
        """Update global state after clearing."""
        self.state[t] = {
            "D_t": D_t,
            "conv_bids": conv_bids,
            "c_t": c_t,
            "q_u": q_u,
            "q_o": q_o,
        }

# ============================================================
# 2. RENEWABLE PRODUCER (Agent R)
# ============================================================

class RenewableProducer:
    """
    Renewable producer deciding quantity and price bids under deep uncertainty.
    """
    def __init__(self, env: MarketEnvironment, controls):
        self.env = env
        self.controls = controls  # dict with {t: (b_t, p_t)}
        self.results = {}

    def profit(self, b_t, p_t, c_t, P_t, q_u, q_o):
        """Hourly profit given realized generation."""
        if p_t > c_t:
            return 0.0
        revenue = c_t * b_t
        penalty_under = q_u * max(b_t - P_t, 0)
        penalty_over = q_o * max(P_t - b_t, 0)
        return revenue - penalty_under - penalty_over

    def evaluate(self, t, c_t, q_u, q_o, P_range):
        """Compute expected and worst-case profit for hour t."""
        b_t, p_t = self.controls[t]
        # Expected profit over price uncertainty (approximated via simulation)
        sim_prices = np.random.normal(c_t, 0.05 * c_t, 100)  # small variance
        exp_profit = 0.0
        for c_sim in sim_prices:
            exp_profit += self.profit(b_t, p_t, c_sim, np.mean(P_range), q_u, q_o)
```

```python
        exp_profit /= len(sim_prices)

        # Worst-case profit over deep uncertainty in P_t
        P_worst = P_range[0] if b_t > np.mean(P_range) else P_range[1]
        worst_profit = self.profit(b_t, p_t, c_t, P_worst, q_u, q_o)

        self.results[t] = {"expected_profit": exp_profit, "worst_profit": worst_profit}
        return exp_profit, worst_profit

# ============================================================
# 3. SYSTEM REGULATOR (Agent S)
# ============================================================

class SystemRegulator:
    """
    Regulator controlling penalty parameters (q_u, q_o) and evaluating system-level outcomes.
    """
    def __init__(self, env: MarketEnvironment, controls):
        self.env = env
        self.controls = controls  # dict of { "q_u": val, "q_o": val }
        self.results = {}

    def compute_imbalance(self, D_t, conv_bids, c_t, b_t, p_t):
        """Compute system imbalance after clearing."""
        accepted_conv = sum(b_i for b_i, p_i in conv_bids if p_i <= c_t)
        accepted_ren = b_t if p_t <= c_t else 0.0
        return D_t - (accepted_conv + accepted_ren)

    def compute_objectives(self, t, D_t, c_t, conv_bids, b_t, p_t):
        """Compute system-level metrics: imbalance, cost, and renewable share."""
        imbalance = abs(self.compute_imbalance(D_t, conv_bids, c_t, b_t, p_t))
        cost = c_t * D_t
        renewable_accepted = b_t if p_t <= c_t else 0.0
        self.results[t] = {
            "imbalance": imbalance,
            "cost": cost,
            "renewable_share": renewable_accepted,
        }
        return imbalance, cost, renewable_accepted

# ============================================================
# 4. UNIFIED SIMULATION FUNCTION (Interface)
# ============================================================

def simulate_market(controls, uncertainties):
    """
    Interface function to simulate the unified model.

    Parameters
    ----------
    controls : dict
        {
          "renewable": {t: (b_t, p_t)},    # Renewable producer bids
          "regulator": {"q_u": ..., "q_o": ...}  # Regulator penalties
        }

    uncertainties : dict
        {
          "P_range": {t: (P_min, P_max)},    # Renewable generation uncertainty ranges
          "D_realizations": list or None,    # Optional pre-specified demand draws
        }

    Returns
    -------
    results : dict
        {
          "Renewable": {"expected_profit": ..., "worst_profit": ...},
          "Regulator": {"avg_imbalance": ..., "avg_cost": ..., "avg_renewable_share": ...}
        }
    """
    # --- Initialize Environment ---
    params = {
        "T": 24,
        "N_c": 3,
        "mu_D": 1000,
```

```
        "sigma_D": 100,
        "mu_b": [400, 350, 300],
        "sigma_b": [50, 40, 30],
        "mu_p": [45, 50, 60],
        "sigma_p": [5, 5, 5],
    }
    env = MarketEnvironment(params, uncertainties)

    # --- Initialize Agents ---
    RP = RenewableProducer(env, controls["renewable"])
    SR = SystemRegulator(env, controls["regulator"])
    q_u, q_o = SR.controls["q_u"], SR.controls["q_o"]

    # --- Simulation loop ---
    results_R, results_S = [], []
    for t in range(1, params["T"] + 1):
        D_t = (uncertainties.get("D_realizations", [None])[t - 1]
               if uncertainties.get("D_realizations") else env.sample_demand(t))
        conv_bids = env.sample_conventional_bids()
        b_t, p_t = RP.controls[t]

        # Market clearing
        c_t = env.market_clearing_price(D_t, conv_bids, b_t, p_t)
        env.update_state(t, D_t, conv_bids, c_t, q_u, q_o)

        # Renewable evaluation
        P_range = uncertainties["P_range"][t]
        exp_profit, worst_profit = RP.evaluate(t, c_t, q_u, q_o, P_range)
        results_R.append((exp_profit, worst_profit))

        # Regulator evaluation
        imbalance, cost, renew_share = SR.compute_objectives(t, D_t, c_t, conv_bids, b_t, p_t)
        results_S.append((imbalance, cost, renew_share))

    # --- Aggregate results ---
    RP_out = {
        "expected_profit": np.mean([r[0] for r in results_R]),
        "worst_profit": np.mean([r[1] for r in results_R]),
    }
    SR_out = {
        "avg_imbalance": np.mean([r[0] for r in results_S]),
        "avg_cost": np.mean([r[1] for r in results_S]),
        "avg_renewable_share": np.mean([r[2] for r in results_S]),
    }

    return {"Renewable": RP_out, "Regulator": SR_out}
```

## ✅ Key Features

- **Modular Design:**
  - **`MarketEnvironment`** handles stochastic draws and market clearing.
  - **`RenewableProducer`** and **`SystemRegulator`** each have self-contained objectives.
- **Shared State:** Market price, demand, and penalties flow through the environment.
- **Exogenous Controls:** Each agent treats the other’s control inputs as external.
- **Quantifiable Outputs:** Returns measurable objectives for both perspectives.

## 🧠 Usage Example (conceptual, not executed)

```python
controls = {
    "renewable": {t: (200, 40 + t % 3) for t in range(1, 25)},
```

```
    "regulator": {"q_u": 15, "q_o": 5},
}

uncertainties = {
    "P_range": {t: (150, 250) for t in range(1, 25)},
}

results = simulate_market(controls, uncertainties)
print(results)
```

This unified modular code provides a clean, **computationally implementable representation** of the composed model — ready for simulation, calibration, or optimization under different stakeholder perspectives.

> Your calculation of worst_profit is incorrect. Since the values of q_u and q_o are different, you should calculate penalty_under and penalty_over for P_range separately. Please refine the implementation.

Excellent — you're absolutely right.
The **worst-case profit** should consider both ends of the uncertain renewable generation range $[\underline{P}_t, \overline{P}_t]$, because the penalties for under- and over-delivery are **asymmetric** (with different $q_u$ and $q_o$).

Therefore, we should compute profits for **both extreme realizations**:

- $P_t = \underline{P}_t$ (possible under-generation → under-delivery penalty $q_u$)
- $P_t = \overline{P}_t$ (possible over-generation → curtailment penalty $q_o$)

and then take the **minimum** of the two profits as the *worst-case* outcome.

Below is the **refined implementation**, with that correction included and a few improvements for clarity and modular structure.

---

```python
import numpy as np

# ============================================================
# 1. SHARED MARKET ENVIRONMENT
# ============================================================

class MarketEnvironment:
    """
    Shared environment representing the day-ahead electricity market.
    Stores stochastic demand, conventional bids, renewable uncertainty,
    and market-clearing logic.
    """
    def __init__(self, params, uncertainties):
        """
        params: dict of fixed parameters (means, variances, time horizon, etc.)
        uncertainties: dict of uncertain variables (e.g., realized demand, conventional bids)
        """
        self.params = params
        self.uncertainties = uncertainties
        self.state = {}

    def sample_demand(self, t):
        mu_D, sigma_D = self.params["mu_D"], self.params["sigma_D"]
        return np.random.normal(mu_D, sigma_D)
```

```python
    def sample_conventional_bids(self):
        """Generate random bids from normal distributions for conventional producers."""
        N_c = self.params["N_c"]
        bids = []
        for i in range(N_c):
            b_i = np.random.normal(self.params["mu_b"][i], self.params["sigma_b"][i])
            p_i = np.random.normal(self.params["mu_p"][i], self.params["sigma_p"][i])
            bids.append((b_i, p_i))
        return bids

    def market_clearing_price(self, D_t, conv_bids, b_t, p_t):
        """
        Solve for clearing price c_t by sorting offers until cumulative supply meets demand.
        """
        offers = conv_bids + [(b_t, p_t)]
        offers.sort(key=lambda x: x[1])  # sort by price ascending
        supply = 0
        for b_i, p_i in offers:
            supply += b_i
            if supply >= D_t:
                return p_i  # clearing price when supply meets demand
        return offers[-1][1]  # fallback: highest offer price

    def update_state(self, t, D_t, conv_bids, c_t, q_u, q_o):
        """Update global state after clearing."""
        self.state[t] = {
            "D_t": D_t,
            "conv_bids": conv_bids,
            "c_t": c_t,
            "q_u": q_u,
            "q_o": q_o,
        }

# ============================================================
# 2. RENEWABLE PRODUCER (Agent R)
# ============================================================

class RenewableProducer:
    """
    Renewable producer deciding quantity and price bids under deep uncertainty.
    """
    def __init__(self, env: MarketEnvironment, controls):
        self.env = env
        self.controls = controls  # dict with {t: (b_t, p_t)}
        self.results = {}

    def profit(self, b_t, p_t, c_t, P_t, q_u, q_o):
        """Hourly profit given realized generation and penalties."""
        if p_t > c_t:
            return 0.0  # bid not accepted
        revenue = c_t * b_t
        penalty_under = q_u * max(b_t - P_t, 0)
        penalty_over = q_o * max(P_t - b_t, 0)
        return revenue - penalty_under - penalty_over

    def evaluate(self, t, c_t, q_u, q_o, P_range):
        """
        Compute expected and worst-case profit for hour t.
        Expected profit: over price uncertainty.
        Worst-case profit: min(profit at P_min, profit at P_max).
        """
        b_t, p_t = self.controls[t]
        P_min, P_max = P_range

        # Simulate expected profit over stochastic prices (simple Monte Carlo)
        sim_prices = np.random.normal(c_t, 0.05 * c_t, 100)
        expected_profit = np.mean([
            self.profit(b_t, p_t, c_sim, np.mean(P_range), q_u, q_o)
            for c_sim in sim_prices
        ])

        # Compute profits at both uncertainty extremes
        profit_low = self.profit(b_t, p_t, c_t, P_min, q_u, q_o)
        profit_high = self.profit(b_t, p_t, c_t, P_max, q_u, q_o)
```

```python
        worst_profit = min(profit_low, profit_high)

        self.results[t] = {
            "expected_profit": expected_profit,
            "worst_profit": worst_profit,
        }
        return expected_profit, worst_profit

# =============================================================
# 3. SYSTEM REGULATOR (Agent S)
# =============================================================

class SystemRegulator:
    """
    Regulator controlling penalty parameters (q_u, q_o) and evaluating system-level outcomes.
    """
    def __init__(self, env: MarketEnvironment, controls):
        self.env = env
        self.controls = controls  # dict of { "q_u": val, "q_o": val }
        self.results = {}

    def compute_imbalance(self, D_t, conv_bids, c_t, b_t, p_t):
        """Compute system imbalance after clearing."""
        accepted_conv = sum(b_i for b_i, p_i in conv_bids if p_i <= c_t)
        accepted_ren = b_t if p_t <= c_t else 0.0
        return D_t - (accepted_conv + accepted_ren)

    def compute_objectives(self, t, D_t, c_t, conv_bids, b_t, p_t):
        """Compute system-level metrics: imbalance, cost, and renewable share."""
        imbalance = abs(self.compute_imbalance(D_t, conv_bids, c_t, b_t, p_t))
        cost = c_t * D_t
        renewable_accepted = b_t if p_t <= c_t else 0.0
        self.results[t] = {
            "imbalance": imbalance,
            "cost": cost,
            "renewable_share": renewable_accepted,
        }
        return imbalance, cost, renewable_accepted

# =============================================================
# 4. UNIFIED SIMULATION FUNCTION (Interface)
# =============================================================

def simulate_market(controls, uncertainties):
    """
    Interface function to simulate the unified model.

    Parameters
    ----------
    controls : dict
        {
          "renewable": {t: (b_t, p_t)},   # Renewable producer bids
          "regulator": {"q_u": ..., "q_o": ...}  # Regulator penalties
        }

    uncertainties : dict
        {
          "P_range": {t: (P_min, P_max)},   # Renewable generation uncertainty ranges
          "D_realizations": list or None,   # Optional pre-specified demand draws
        }

    Returns
    -------
    results : dict
        {
          "Renewable": {"expected_profit": ..., "worst_profit": ...},
          "Regulator": {"avg_imbalance": ..., "avg_cost": ..., "avg_renewable_share": ...}
        }
    """
    # --- Initialize Environment ---
    params = {
        "T": 24,
        "N_c": 3,
        "mu_D": 1000,
        "sigma_D": 100,
```

```
        "mu_b": [400, 350, 300],
        "sigma_b": [50, 40, 30],
        "mu_p": [45, 50, 60],
        "sigma_p": [5, 5, 5],
    }
    env = MarketEnvironment(params, uncertainties)

    # --- Initialize Agents ---
    RP = RenewableProducer(env, controls["renewable"])
    SR = SystemRegulator(env, controls["regulator"])
    q_u, q_o = SR.controls["q_u"], SR.controls["q_o"]

    # --- Simulation loop ---
    results_R, results_S = [], []
    for t in range(1, params["T"] + 1):
        D_t = (uncertainties.get("D_realizations", [None])[t - 1]
               if uncertainties.get("D_realizations") else env.sample_demand(t))
        conv_bids = env.sample_conventional_bids()
        b_t, p_t = RP.controls[t]

        # Market clearing
        c_t = env.market_clearing_price(D_t, conv_bids, b_t, p_t)
        env.update_state(t, D_t, conv_bids, c_t, q_u, q_o)

        # Renewable evaluation
        P_range = uncertainties["P_range"][t]
        exp_profit, worst_profit = RP.evaluate(t, c_t, q_u, q_o, P_range)
        results_R.append((exp_profit, worst_profit))

        # Regulator evaluation
        imbalance, cost, renew_share = SR.compute_objectives(t, D_t, c_t, conv_bids, b_t, p_t)
        results_S.append((imbalance, cost, renew_share))

    # --- Aggregate results ---
    RP_out = {
        "expected_profit": np.mean([r[0] for r in results_R]),
        "worst_profit": np.mean([r[1] for r in results_R]),
    }
    SR_out = {
        "avg_imbalance": np.mean([r[0] for r in results_S]),
        "avg_cost": np.mean([r[1] for r in results_S]),
        "avg_renewable_share": np.mean([r[2] for r in results_S]),
    }

    return {"Renewable": RP_out, "Regulator": SR_out}
```

---

## ✅ Refinements Applied

- The **worst-case profit** now correctly considers **both** $P_{\min}$ and $P_{\max}$:

  ```python
  profit_low = self.profit(b_t, p_t, c_t, P_min, q_u, q_o)
  profit_high = self.profit(b_t, p_t, c_t, P_max, q_u, q_o)
  worst_profit = min(profit_low, profit_high)
  ```

- This accounts for asymmetric penalties $q_u \neq q_o$.
- The code remains **modular**, **quantitative**, and **ready for simulation or optimization**.

---

## 🧠 Example (conceptual, not executed)

```python
controls = {
    "renewable": {t: (200, 42 + (t % 4)) for t in range(1, 25)},
    "regulator": {"q_u": 20, "q_o": 5},
}

uncertainties = {
    "P_range": {t: (150, 250) for t in range(1, 25)},
}

results = simulate_market(controls, uncertainties)
print(results)
```

This version is now **fully internally consistent**, **implementable**, and properly handles **asymmetric penalties** in the worst-case profit calculation.