

Experiments 3 - Market_narrative 1



In this problem, a renewable energy producer participates in a day-ahead merit-order electricity market competing against three other conventional producers. The producer submits hourly bids that specify a delivery quantity of electricity and an offer price for each hour of the next day. 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. Meanwhile, the hourly clearing price depends on the bids of all producers and is unknown in advance. The renewable energy producer's goal is to maximize its expected profit by strategically selecting the quantity of electricity b_t and offer price p_t to bid for each hour of the next day, accounting for the uncertainty in both renewable generation and clearing prices.

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.

Excellent — this is a classic **stochastic bidding optimization** problem in a day-ahead electricity market under **dual uncertainty**: renewable output and market clearing price. Let's formalize it step by step as a **stochastic decision model**.

1. Time structure

- Discrete time: $t = 1, 2, \dots, T$ (hours of the next day).

2. Decision variables (control variables)

At each hour t :

- $b_t \in \mathbb{R}_+$: **bid quantity** (MWh) of electricity offered.
- $p_t \in \mathbb{R}_+$: **offer price** (\$/MWh).

These are chosen **before** the uncertainties (renewable generation and market price) are realized.

3. Random (stochastic) variables

Two key sources of uncertainty for each hour t :

1. Renewable generation availability

$$G_t : \Omega \rightarrow [0, \bar{G}_t]$$

where G_t is the realized renewable generation capacity (MWh) — random due to weather (wind, solar, etc.).

2. Market clearing price

$$P_t^c : \Omega \rightarrow [0, \infty)$$

the day-ahead clearing price determined by the intersection of aggregate supply and demand. It depends on all market participants' bids.

Both G_t and P_t^c are **unknown** when bidding decisions are made, but probability distributions (or scenario forecasts) may be available.

4. Outcome of market participation

For each hour t :

- The bid (b_t, p_t) is **accepted** if the offer price is below or equal to the clearing price:

$$\text{Acceptance indicator: } I_t = \mathbf{1}_{\{p_t \leq P_t^c\}}.$$

- If accepted, the producer must deliver b_t MWh.
- If actual generation $G_t < b_t$, the producer must buy the shortfall $b_t - G_t$ from the spot (or balancing) market at a penalty price P_t^{pen} .

5. Revenue and cost functions

The **realized profit** at time t is:

$$\pi_t = \begin{cases} P_t^c \cdot b_t - P_t^{\text{pen}} \cdot \max(0, b_t - G_t), & \text{if } p_t \leq P_t^c, \\ 0, & \text{otherwise.} \end{cases}$$

6. Objective function

The renewable producer wants to **maximize expected total profit** over the day:

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

Expectations are taken with respect to the joint distribution of (G_t, P_t^c) , possibly correlated across time.

7. Constraints

- Capacity limits:

$$0 \leq b_t \leq \bar{G}_t.$$

- Price bounds (e.g., market limits):

$$0 \leq p_t \leq \bar{P}.$$

- Possibly, intertemporal or operational constraints (optional):
 - e.g., ramping or energy storage balance if coupled to a battery.
-

8. State variables

If the problem is **multi-period coupled** (e.g., with storage, cumulative penalties, or learning), we can define:

- S_t : state at hour t , possibly including:
 - stored energy (if hybrid with battery),
 - cumulative profit,
 - information set (forecast updates).

Otherwise, if decisions are independent by hour, there is no state transition: $S_{t+1} = S_t$.

9. Transition functions

If dynamic:

$$S_{t+1} = f(S_t, b_t, G_t, P_t^c),$$

where f updates the system state (e.g., energy storage level, forecast beliefs, etc.).

In the simple independent-hour model, f is trivial (no intertemporal dependence).

10. Uncertainty model

- Joint probability distributions:

$$(G_t, P_t^c) \sim \mathcal{P}_t,$$

possibly scenario-based or represented by samples.

- These may be modeled as:
 - independent across t , or
 - correlated with known conditional forecasts $\mathbb{P}(G_t, P_t^c | \text{forecast data})$.
-

11. Compact model summary

Stochastic Optimization Model:

$$\begin{aligned} \max_{\{b_t, p_t\}} \quad & \mathbb{E} \left[\sum_{t=1}^T \mathbf{1}_{\{p_t \leq P_t^c\}} (P_t^c b_t - P_t^{\text{pen}}(b_t - G_t)^+) \right] \\ \text{s.t.} \quad & 0 \leq b_t \leq \bar{G}_t, \quad 0 \leq p_t \leq \bar{P}, \quad \forall t. \end{aligned}$$

12. Key components summary table

Component	Symbol	Description
Decision variables	b_t, p_t	Bid quantity and offer price
Random variables	G_t, P_t^c	Renewable generation, clearing price
Parameters	$\bar{G}_t, \bar{P}, P_t^{\text{pen}}$	Max capacity, max price, penalty
State variables	S_t (optional)	System state (e.g. storage, info)
Transition	$S_{t+1} = f(S_t, b_t, G_t, P_t^c)$	System evolution rule
Profit function	$\pi_t(b_t, p_t, G_t, P_t^c)$	Hourly profit realization
Objective	$\max \mathbb{E}[\sum_t \pi_t]$	Maximize expected total profit

Would you like me to **extend this to a scenario-based (two-stage) stochastic programming formulation** — e.g., with discrete scenarios for (G_t, P_t^c) ? That's the standard next step to make it computationally solvable.



First, the day-ahead market is cleared once per day for 24 hourly intervals of the following day, so you can assume the problem only considers a 24-hour period. Second, penalties would apply if the renewable producer under-delivers or over-delivers relative to its bid. Please refine the specification.

Excellent — that's a very clear and realistic refinement.

We'll now formalize the model for **a single day (24 hours)**, with **dual-sided penalties** for deviations (both under-delivery and over-delivery) relative to the day-ahead bid.

Below is the refined, fully specified **stochastic bidding model**.

1. Time structure

- The market clears once per day for 24 hourly intervals:
 $t = 1, 2, \dots, 24$.

All bids (b_t, p_t) for each hour t are submitted simultaneously **before** uncertainties (generation and market price) are realized.

2. Decision (control) variables

For each hour t :

- $b_t \geq 0$: quantity of electricity bid into the day-ahead market (MWh).
- $p_t \geq 0$: offer price for that quantity (\$/MWh).

These are chosen **before** the market clearing and weather outcomes are known.

3. Stochastic (random) variables

Two key uncertainties per hour:

- G_t : **available renewable generation** (MWh).
Random due to weather; $G_t \in [0, \bar{G}_t]$.
- P_t^c : **market clearing price** (\$/MWh).
Determined by the market operator based on all participants' bids and system demand.

Both are random variables with a joint (possibly correlated) probability distribution known through forecasts:

$$(G_t, P_t^c) \sim \mathcal{P}_t.$$

4. Market clearing and delivery outcome

The bid (b_t, p_t) is **accepted** if the offer price does not exceed the clearing price:

$$I_t = \mathbf{1}_{\{p_t \leq P_t^c\}}.$$

If $I_t = 1$, the producer is **scheduled** to deliver b_t MWh at the clearing price P_t^c .

After renewable generation G_t is realized, deviations occur:

- **Under-delivery**: if $G_t < b_t$, shortfall $b_t - G_t$.
- **Over-delivery**: if $G_t > b_t$, surplus $G_t - b_t$.

Both deviations incur settlement penalties or imbalance prices.

5. Penalty (settlement) prices

Let:

- P_t^u : penalty (or buy-back) price for under-delivery (\$/MWh).
Typically $P_t^u > P_t^c$ (since buying from the balancing market is costly).
- P_t^o : penalty (or sell-back) price for over-delivery (\$/MWh).
Typically $P_t^o < P_t^c$ (since surplus energy is sold at a discount).

These parameters are known constants or random with known expectations.

6. Hourly realized profit

For hour t :

$$\pi_t = I_t \left[P_t^c b_t - P_t^u (b_t - G_t)^+ + P_t^o (G_t - b_t)^+ \right],$$

where:

- $(x)^+ = \max(0, x)$,
- if the bid is not accepted ($I_t = 0$), $\pi_t = 0$.

7. Objective function

The renewable producer maximizes **expected daily profit**:

$$\max_{\{b_t, p_t\}_{t=1}^{24}} \mathbb{E} \left[\sum_{t=1}^{24} I_t \left(P_t^c b_t - P_t^u (b_t - G_t)^+ + P_t^o (G_t - b_t)^+ \right) \right].$$

Expectation is over the joint distribution of all random variables $(G_t, P_t^c)_{t=1}^{24}$.

8. Constraints

- Renewable capacity:

$$0 \leq b_t \leq \bar{G}_t, \quad \forall t.$$

- Offer price bounds:

$$0 \leq p_t \leq \bar{P}, \quad \forall t.$$

- (Optional) correlations or operational couplings, e.g.:

- ramp-rate or temporal consistency in bids,
- forecast coupling across hours.

If each hour is independent, the problem can decompose by t .

9. Model components summary

Category	Symbol	Description
Decision variables	b_t, p_t	Bid quantity and offer price for hour t
Random variables	G_t, P_t^c	Realized generation and market clearing price
Parameters	$\bar{G}_t, \bar{P}, P_t^u, P_t^o$	Max capacity, price bounds, penalty prices
Indicator	$I_t = \mathbf{1}_{\{p_t \leq P_t^c\}}$	Bid acceptance indicator

Category	Symbol	Description
Profit function	π_t	Hourly realized profit
Objective	$\max \mathbb{E}[\sum_t \pi_t]$	Expected total daily profit
Constraints	$0 \leq b_t \leq \bar{G}_t, 0 \leq p_t \leq \bar{P}$	Physical and market limits

10. Optional dynamic structure (if coupled)

If forecasting or learning evolves during the day, define:

- **State variable:** S_t = information set (forecast updates or cumulative imbalance).
- **Transition:** $S_{t+1} = f(S_t, G_t, P_t^c)$.
- Decisions could then depend on S_t : $(b_t, p_t) = \pi(S_t)$.

For a standard 24-hour day-ahead problem, however, all bids are simultaneous and **no intertemporal transition** occurs within the day.

✓ Final compact model

$$\begin{aligned} & \max_{\{b_t, p_t\}_{t=1}^{24}} \mathbb{E} \left[\sum_{t=1}^{24} \mathbf{1}_{\{p_t \leq P_t^c\}} (P_t^c b_t - P_t^u (b_t - G_t)^+ + P_t^o (G_t - b_t)^+) \right] \\ & \text{s.t. } 0 \leq b_t \leq \bar{G}_t, \quad 0 \leq p_t \leq \bar{P}, \quad t = 1, \dots, 24. \end{aligned}$$

Would you like me to **express this in a two-stage stochastic programming form** next (e.g. scenario set for (G_t, P_t^c) , first-stage bids, second-stage realized penalties)? That's the usual structure for numerical optimization.



There are still several areas where the model specification can be improved: (1) the renewable energy generation is deeply uncertain, so its probability distribution cannot be determined; and (2) the market-clearing price is not random but endogenous, determined by the merit-order of bids. Please refine your specification.

Perfect — this refinement goes to the **core modeling philosophy** of electricity market participation by renewables.

You're absolutely right:

- Renewable generation is **deeply uncertain**, i.e. no well-defined probability law (so we cannot simply "take expectations").
- The **market-clearing price P_t^c** is **endogenous**, determined by the intersection of aggregate supply (including our bid) and demand.

We now move from a **stochastic optimization** to a **robust-equilibrium** (or **bilevel**) formulation.
Let's build this systematically.

1. Time structure

Day-ahead market: 24 discrete hourly intervals for the next day:

$$t = 1, 2, \dots, 24.$$

All bids for the day are submitted simultaneously and jointly determine the market clearing.

2. Decision variables (renewable producer's bids)

For each hour t :

- $b_t \geq 0$: bid quantity (MWh).
- $p_t \geq 0$: offer price (\$/MWh).

The vector of bids $\{(b_t, p_t)\}_{t=1}^{24}$ constitutes the renewable producer's decision.

3. Uncertain renewable generation (deep uncertainty)

Renewable output for hour t :

$$G_t \in \mathcal{G}_t \subseteq [0, \bar{G}_t],$$

where:

- \mathcal{G}_t is a **set of plausible realizations**, not a probabilistic distribution;
- e.g. defined by forecast bounds or scenarios:

$$\mathcal{G}_t = \{G_t : \underline{G}_t \leq G_t \leq \bar{G}_t\}.$$

This reflects **deep uncertainty** — we cannot assign probabilities, so we aim for **robust (worst-case)** or **distributionally robust** decisions.

4. Market clearing (endogenous prices)

Let there be N conventional producers indexed by $i = 1, \dots, N$ and the renewable producer indexed by r .

Each producer i submits a bid $(b_{i,t}, p_{i,t})$.

The market operator clears the market for each hour t by solving a **merit-order dispatch**:

$$\begin{aligned}
 & \text{Find } P_t^c, \{q_{i,t}\}_{i \in \mathcal{I} \cup \{r\}} \\
 \text{s.t. } & \sum_{i \in \mathcal{I} \cup \{r\}} q_{i,t} = D_t, \\
 & 0 \leq q_{i,t} \leq b_{i,t}, \\
 & q_{i,t} > 0 \Rightarrow p_{i,t} \leq P_t^c, \quad q_{i,t} < b_{i,t} \Rightarrow p_{i,t} \geq P_t^c.
 \end{aligned}$$

This defines the **equilibrium price** P_t^c — the marginal accepted offer.

Hence, P_t^c is **endogenous**, depending on all bids, including the renewable's (b_t, p_t) .

5. Delivery and penalties

Once the market clears:

- The renewable producer is **scheduled** to deliver b_t^* MWh (its accepted quantity — equal to its bid if accepted).
- Actual generation G_t realizes.
- Deviation:

$$\Delta_t = G_t - b_t^*.$$

- Settlement penalties:

$$\text{Penalty cost} = P_t^u \cdot (\Delta_t^-) + P_t^o \cdot (\Delta_t^+),$$

$$\text{where } \Delta_t^- = \max(0, -\Delta_t), \Delta_t^+ = \max(0, \Delta_t).$$

Under-delivery (shortfall) costs P_t^u per MWh; over-delivery (surplus) is compensated at P_t^o per MWh.

6. Hourly profit function (conditional on clearing)

Given a clearing outcome (P_t^c, b_t^*) and a realization G_t :

$$\pi_t = P_t^c b_t^* - P_t^u (b_t^* - G_t)^+ + P_t^o (G_t - b_t^*)^+.$$

Note that b_t^* itself is determined by the market-clearing process, hence depends implicitly on (b_t, p_t) and others' bids.

7. Renewable producer's problem: robust bilevel formulation

Since P_t^c is determined by market clearing, the renewable producer's **upper-level problem** is:

$$\begin{aligned}
& \max_{\{b_t, p_t\}_{t=1}^{24}} \min_{G_t \in \mathcal{G}_t, t=1, \dots, 24} \sum_{t=1}^{24} \left[P_t^c(b_t, p_t, \mathbf{b}_{-r,t}, \mathbf{p}_{-r,t}) b_t^* - P_t^u(b_t^* - G_t)^+ + P_t^o(G_t - b_t^*)^+ \right] \\
& \text{s.t. } 0 \leq b_t \leq \bar{G}_t, \quad 0 \leq p_t \leq \bar{P}, \quad \forall t, \\
& (P_t^c, \{q_{i,t}\}) \text{ satisfy market clearing constraints (as above).}
\end{aligned}$$

- The **inner minimization** over G_t reflects **robustness** to deep uncertainty: the renewable seeks to maximize the minimum possible profit over all feasible generation realizations.
- The market-clearing mechanism defines **endogenous prices** and accepted quantities through a **lower-level equilibrium** problem.

8. Alternative formulation: distributionally robust (if partial information)

If limited forecast information (e.g. mean, bounds) on G_t exists, a **distributionally robust** expectation can replace the inner minimum:

$$\max_{\{b_t, p_t\}} \min_{\mathbb{P} \in \mathcal{D}} \mathbb{E}_{\mathbb{P}}[\Pi(\{b_t, p_t\}, G, P^c)],$$

where \mathcal{D} is an ambiguity set of distributions consistent with known forecast statistics.

9. Key model components (refined)

Category	Symbol	Description
Decision variables	b_t, p_t	Renewable's bid quantity and offer price
Uncertain variables (deep uncertainty)	$G_t \in \mathcal{G}_t$	Renewable generation realizations (no fixed probability)
Endogenous variables	$P_t^c, q_{i,t}$	Market clearing price and dispatch quantities (determined by all bids)
Parameters	$D_t, P_t^u, P_t^o, \bar{G}_t, \bar{P}$	Demand, penalty rates, physical limits
State variables	None (single-day, simultaneous bidding)	
Transition	Not applicable (static one-shot market)	
Objective	$\max \min_{G_t \in \mathcal{G}_t} \sum \pi_t$	Robust expected-profit maximization
Structure	Bilevel (upper-level: bidding; lower-level: market clearing)	Strategic optimization with endogenous prices

10. Interpretation

- The renewable producer's problem is **strategic** (game-theoretic) and **robust** (non-stochastic).
- The market-clearing subproblem embeds **merit-order dispatch** determining prices and accepted quantities.

- Deep uncertainty in G_t is treated **via a set-based (min-max)** approach.
- This structure generalizes stochastic bidding models to settings with **unknown distributions and endogenous prices**, i.e. **robust bilevel energy market optimization**.

Would you like me to write this model in **explicit bilevel optimization form** (with KKT conditions for market clearing) — suitable for conversion to a single-level MILP or complementarity formulation for computational analysis?



You can assume the total energy demand D_t follows a normal distribution with constant mean and variance. Furthermore, there are three conventional producers, and you can also assume that 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 the model specification.

Excellent — now we can complete a **fully specified formal model** that integrates:

1. **Deep uncertainty** in renewable generation,
2. **Endogenous market-clearing prices** determined by merit-order, and
3. **Probabilistic uncertainty** in total demand and conventional producers' bids.

The resulting model is a **hybrid robust-stochastic bilevel optimization** problem for the renewable energy producer's strategic bidding.

Let's carefully structure it step-by-step.

1. Time structure

Day-ahead market, 24 hourly intervals:

$$t = 1, 2, \dots, 24.$$

All bids are submitted simultaneously for the next day.

2. Decision (control) variables

Renewable producer r chooses, for each hour t :

- $b_t \geq 0$: bid quantity (MWh).
- $p_t \geq 0$: offer price (\$/MWh).

These are the **first-stage decisions** (made before any uncertainty realization).

3. Uncertain and random variables

Type	Symbol	Description	Distribution / Uncertainty type
Deeply uncertain	G_t	Available renewable generation (MWh)	Unknown distribution; only bounded support $G_t \in [\underline{G}_t, \bar{G}_t]$
Random	D_t	Total demand (MWh)	$D_t \sim \mathcal{N}(\mu_D, \sigma_D^2)$
Random	$(b_{it}, p_{it}), i = 1, 2, 3$	Bids of conventional producers	$b_{it} \sim \mathcal{N}(\mu_{b_i}, \sigma_{b_i}^2), p_{it} \sim \mathcal{N}(\mu_{p_i}, \sigma_{p_i}^2)$
Endogenous	P_t^c	Market-clearing price	Determined by merit-order of all producers' bids

Thus:

- G_t = deeply uncertain (worst-case treatment);
- $D_t, (b_{it}, p_{it})$ = stochastic (probabilistic);
- P_t^c = endogenous (market-clearing outcome).

4. Market clearing mechanism (lower-level problem)

For each hour t , given all bids:

$$\{(b_t, p_t)\} \cup \{(b_{it}, p_{it}), i = 1, 2, 3\},$$

and realized demand D_t ,

the market operator determines dispatch quantities q_{it} and the clearing price P_t^c by solving a **merit-order dispatch**:

$$\begin{aligned} & \min_{\{q_{it}\}, q_t^r, P_t^c} \quad \sum_{i=1}^3 p_{it} q_{it} + p_t q_t^r \\ & \text{s.t.} \quad \sum_{i=1}^3 q_{it} + q_t^r = D_t, \\ & \quad 0 \leq q_{it} \leq b_{it}, \quad 0 \leq q_t^r \leq b_t, \\ & \quad q_{it}(p_{it} - P_t^c) = 0, \quad (b_{it} - q_{it})(P_t^c - p_{it}) = 0, \\ & \quad q_t^r(p_t - P_t^c) = 0, \quad (b_t - q_t^r)(P_t^c - p_t) = 0. \end{aligned}$$

This formulation enforces **market equilibrium conditions**:

- The clearing price P_t^c is the marginal accepted bid;
- Each producer's dispatch q_{it} or q_t^r lies within its bid capacity.

5. Real-time delivery and penalties

After market clearing:

- Renewable producer is **scheduled** to deliver q_t^r MWh.
- Actual available generation G_t realizes (within the uncertain set).

- The deviation:

$$\Delta_t = G_t - q_t^r.$$

- Penalty structure (settlement):

$$\text{Penalty}_t = P_t^u (\Delta_t^-) + P_t^o (\Delta_t^+),$$

where:

- $\Delta_t^- = \max(0, -\Delta_t)$ (under-delivery),
- $\Delta_t^+ = \max(0, \Delta_t)$ (over-delivery),
- $P_t^u > P_t^c > P_t^o$.

6. Hourly realized profit

Given clearing outcome (q_t^r, P_t^c) and realized G_t :

$$\pi_t = P_t^c q_t^r - P_t^u (q_t^r - G_t)^+ + P_t^o (G_t - q_t^r)^+.$$

7. Renewable producer's robust-stochastic bilevel optimization

Upper-level (renewable producer):

$$\begin{aligned} \max_{\{b_t, p_t\}_{t=1}^{24}} \quad & \mathbb{E}_{D_t, (b_{it}, p_{it})} \left[\min_{G_t \in [\underline{G}_t, \bar{G}_t]} \sum_{t=1}^{24} \pi_t(b_t, p_t, G_t, P_t^c) \right] \\ \text{s.t.} \quad & 0 \leq b_t \leq \bar{G}_t, \quad 0 \leq p_t \leq \bar{P}, \quad \forall t, \\ & (P_t^c, q_t^r, q_{it}) \text{ satisfy market-clearing conditions.} \end{aligned}$$

Lower-level (market clearing):

For each t ,

$$(P_t^c, q_t^r, q_{it}) = \operatorname{argmin}_{\{q_{it}\}, q_t^r, P_t^c} \left\{ \sum_{i=1}^3 p_{it} q_{it} + p_t q_t^r : \sum q_{it} + q_t^r = D_t, 0 \leq q_{it} \leq b_{it}, 0 \leq q_t^r \leq b_t \right\}.$$

The expectation $\mathbb{E}_{D_t, (b_{it}, p_{it})}$ integrates over the random market environment (demand and competitors' bids), while the inner minimization over G_t captures robustness to deep renewable uncertainty.

8. Interpretation of layers

Layer	Random / Deterministic	Description
Outer layer (renewable)	Deterministic decision	Chooses bids to maximize robust expected profit

Layer	Random / Deterministic	Description
Middle layer (market clearing)	Endogenous equilibrium	Determines clearing price and accepted quantity
Inner layer (renewable generation)	Adversarial uncertainty	Nature chooses worst-case $G_t \in [\underline{G}_t, \bar{G}_t]$
Expectation layer	Probabilistic	Integrates over random demand and competitors' bids

9. Compact model representation

$$\begin{aligned}
& \max_{\{b_t, p_t\}_{t=1}^{24}} \mathbb{E}_{D_t, (b_{it}, p_{it})} \left[\min_{G_t \in [\underline{G}_t, \bar{G}_t]} \sum_{t=1}^{24} \left(P_t^c q_t^r - P_t^u (q_t^r - G_t)^+ + P_t^o (G_t - q_t^r)^+ \right) \right] \\
& \text{s.t. } 0 \leq b_t \leq \bar{G}_t, \quad 0 \leq p_t \leq \bar{P}, \\
& \quad (P_t^c, q_t^r, q_{it}) \in \text{MarketClearing}(D_t, \{b_{it}, p_{it}\}, b_t, p_t).
\end{aligned}$$

10. Model components summary

Category	Symbol	Type	Description
Decision variables	b_t, p_t	Control	Renewable bid quantity and offer price
Random variables	D_t, b_{it}, p_{it}	Stochastic	Demand and competitors' bids, normally distributed
Uncertain variables	G_t	Deep uncertainty	Renewable generation, set-valued uncertainty
Endogenous variables	P_t^c, q_t^r, q_{it}	Determined by equilibrium	Clearing price and dispatched quantities
Parameters	$\mu_D, \sigma_D, \mu_{b_i}, \sigma_{b_i}, \mu_{p_i}, \sigma_{p_i}, P_t^u, P_t^o, \bar{G}_t, \bar{P}$	Known constants	
Objective	—	Hybrid robust-stochastic	Maximize expected profit under worst-case renewable realization
Structure	—	Bilevel	Upper: renewable's strategic bidding; Lower: market clearing equilibrium

Conceptual Summary

This model represents:

- **Stochastic environment:** driven by normally distributed **demand** and **conventional bids**, which define a random market environment.
- **Endogenous prices:** arise from merit-order clearing among all bids (renewable and conventional).

- **Deep uncertainty:** renewable generation modeled through a **robust set** rather than a probability distribution.
- **Strategic decision:** renewable producer optimizes bids to **maximize robust expected profit** accounting for both uncertain generation and uncertain market response.

Would you like me to take the next step and **express this model as a single-level equivalent**, for example by replacing the lower-level market-clearing problem with its KKT (complementarity) conditions, to make it computationally solvable as an MPEC or MILP?



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'll move from a single-agent optimization (the renewable producer's problem) to a **multi-stakeholder system specification** where:

1. Both the **renewable producer** and the **system regulator** act as independent agents;
2. They share a **common environment** (the day-ahead market, demand, competitors, and uncertainty); and
3. Their decisions **interact indirectly** through this shared environment — market clearing, prices, and penalties.

Below is a **refined, structured system model** showing how both agents operate within the same environment.

I. Common Environment (Shared by All Stakeholders)

1. Time

- The system operates for a **24-hour day-ahead market horizon**:
 $t = 1, 2, \dots, 24$.

2. Global Parameters (known to all agents)

Symbol	Description
\bar{G}_t	Renewable producer's maximum generation capacity (MWh)
\bar{P}	Maximum allowable bid price (\$/MWh)

Symbol	Description
P_t^u, P_t^o	Penalty prices for under- and over-delivery
μ_D, σ_D^2	Mean and variance of total demand D_t
$\mu_{b_i}, \sigma_{b_i}^2$	Mean and variance of bid quantities of conventional producer i
$\mu_{p_i}, \sigma_{p_i}^2$	Mean and variance of offer prices of conventional producer i
G_t, \bar{G}_t	Bounds on renewable generation (deep uncertainty set)

3. Global Random / Uncertain Variables

Symbol	Type	Description
$D_t \sim \mathcal{N}(\mu_D, \sigma_D^2)$	Random	Total demand (MWh)
$(b_{it}, p_{it}) \sim \mathcal{N}(\mu_{b_i}, \sigma_{b_i}^2) \times \mathcal{N}(\mu_{p_i}, \sigma_{p_i}^2)$	Random	Conventional producers' bids
$G_t \in [G_t, \bar{G}_t]$	Deep uncertainty	Renewable generation realization (MWh)

4. Global Endogenous Variables (determined by market clearing)

Symbol	Description
P_t^c	Market-clearing price (\$/MWh)
q_t^r	Accepted renewable quantity (MWh)
q_{it}	Accepted quantity of conventional producer i (MWh)

These endogenous variables are **shared** across both stakeholders: both the renewable producer and the regulator influence them indirectly through their decisions.

5. Shared Transition Function (market equilibrium)

For each hour t :

$$\text{MarketClearing: } \begin{cases} \sum_{i=1}^3 q_{it} + q_t^r = D_t, \\ 0 \leq q_{it} \leq b_{it}, \quad 0 \leq q_t^r \leq b_t, \\ \text{Complementarity (merit-order):} \\ q_{it}(p_{it} - P_t^c) = 0, \quad (b_{it} - q_{it})(P_t^c - p_{it}) = 0, \\ q_t^r(p_t - P_t^c) = 0, \quad (b_t - q_t^r)(P_t^c - p_t) = 0. \end{cases}$$

This **market equilibrium mechanism** defines the shared system dynamics.

II. Perspective 1: Renewable Producer

1. Control Variables

For each hour t :

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

Chosen simultaneously for the next day (first-stage decision).

2. State and Transition

Since all bids are submitted simultaneously for the 24-hour horizon, there are no inter-hour state transitions.

However, if an extension includes storage or adaptive forecasting, the state could be $S_t =$ (forecast info, storage level).

For this static setting, transitions are trivial.

3. Exogenous Uncertainties

- Random demand D_t and conventional bids (b_{it}, p_{it}) ;
- Deeply uncertain renewable generation $G_t \in [\underline{G}_t, \bar{G}_t]$.

4. Endogenous Feedback from the Environment

- The producer's bid (b_t, p_t) affects the **clearing price** P_t^c and **accepted quantity** q_t^r via market clearing.
- These determine the producer's realized profit.

5. Objective Functions (multi-objective view)

The renewable producer may have **two main objectives**:

(a) Economic Objective (profit maximization)

$$\max_{b_t, p_t} \mathbb{E}_{D_t, (b_{it}, p_{it})} \left[\min_{G_t \in [\underline{G}_t, \bar{G}_t]} \sum_{t=1}^{24} \left(P_t^c q_t^r - P_t^u (q_t^r - G_t)^+ + P_t^o (G_t - q_t^r)^+ \right) \right].$$

(b) Reliability Objective (minimize imbalance)

$$\min_{b_t, p_t} \mathbb{E}_{D_t, (b_{it}, p_{it})} \left[\max_{G_t \in [\underline{G}_t, \bar{G}_t]} \sum_{t=1}^{24} |G_t - q_t^r| \right].$$

This reflects a secondary goal of maintaining supply reliability and reducing penalties.

6. Constraints

$$0 \leq b_t \leq \bar{G}_t, \quad 0 \leq p_t \leq \bar{P}, \quad \forall t.$$

7. Renewable Producer Model Summary

Component	Symbol / Equation	Description
Control vars	b_t, p_t	Bidding decisions
Random vars	$D_t, (b_{it}, p_{it})$	Market uncertainties
Deep uncertainty	$G_t \in [\underline{G}_t, \bar{G}_t]$	Renewable generation
Transition	MarketClearing equations	Coupling with system
Objectives	(1) Expected robust profit (2) Reliability	Multi-objective formulation
Coupling	via P_t^c, q_t^r	Endogenous price and allocation

III. Perspective 2: System Regulator

The **system regulator** oversees market design and reliability.

They **do not control bids directly**, but can influence the system through **policy and regulatory parameters** that shape incentives and outcomes.

1. Control Variables (policy levers)

For each hour t :

- P_t^u, P_t^o : penalty coefficients for under/over delivery;
- Possibly \bar{P} : price cap;
- Market design parameters (e.g., tolerance bands, renewable quotas, or reserve margin requirements).

Let

$$\theta = \{P_t^u, P_t^o, \bar{P}\}_{t=1}^{24}$$

represent the **regulator's control vector**.

2. State and Transition

The regulator observes and evolves the **system state**:

$$S_t^{\text{sys}} = (D_t, \{b_{it}, p_{it}\}, b_t, p_t, P_t^c, q_t^r),$$

with transitions driven by the **market-clearing mechanism** and random realizations of D_t and G_t .

3. Objectives (multi-objective view)

(a) Market Efficiency (economic surplus)

Maximize expected social welfare:

$$\max_{\theta} \mathbb{E} \left[\sum_{t=1}^{24} (\text{ConsumerSurplus}(D_t, P_t^c) + \text{ProducerSurplus}(P_t^c, q_{it}, q_t^r)) \right].$$

Consumer and producer surpluses depend on clearing price and allocations.

(b) System Reliability

Minimize expected total imbalance (both over- and under-delivery):

$$\min_{\theta} \mathbb{E} \left[\sum_{t=1}^{24} |G_t - q_t^r| \right].$$

(c) Fairness / Market Stability

Maintain equitable price signals and bounded volatility:

$$\min_{\theta} \text{Var}(P_t^c).$$

4. Constraints

- Regulatory feasibility (e.g., penalties within legal bounds):

$$0 \leq P_t^o \leq P_t^c \leq P_t^u \leq \bar{P}.$$

- Reliability thresholds:

$$\Pr(|G_t - q_t^r| > \epsilon) \leq \alpha, \quad \forall t.$$

5. Regulator Model Summary

Component	Symbol / Equation	Description
Control vars	P_t^u, P_t^o, \bar{P}	Policy levers influencing incentives
Random vars	$D_t, (b_{it}, p_{it}), G_t$	Market and physical uncertainties
Transition	MarketClearing + Producer reactions	Coupled system evolution
Objectives	(1) Maximize social welfare (2) Minimize imbalance (3) Minimize price volatility	Multi-objective
Coupling	via P_t^c, q_t^r	Endogenous market outcomes influenced by policy

IV. Coupled Multi-Agent System

The two perspectives share a **common environment**:

Shared Variables	Controlled by	Affected Stakeholders
P_t^c, q_t^r, q_{it}	Market equilibrium	Both

Shared Variables	Controlled by	Affected Stakeholders
$G_t, D_t, (b_{it}, p_{it})$	Nature / external	Both
P_t^u, P_t^o	Regulator	Producer, market
b_t, p_t	Renewable producer	Regulator (through system reliability)

The interaction is **indirect and feedback-based**:

- The renewable producer's bids affect P_t^c and imbalance outcomes.
- The regulator's penalties and rules affect the producer's optimal bidding strategy.

V. System Summary (Formal Structure)

Element	Renewable Producer	System Regulator	Shared Environment
Decision Variables	b_t, p_t	P_t^u, P_t^o, \bar{P}	P_t^c, q_t^r, q_{it}
Uncertainty	G_t (deep), $D_t, (b_{it}, p_{it})$	same	same
Transition / Coupling	MarketClearing equations	MarketClearing equations	Defines P_t^c, q_t^r
Objectives	Profit maximization, reliability	Welfare, reliability, stability	Jointly depend on equilibrium
Type of Problem	Robust-stochastic optimization	Policy optimization (multi-objective)	Endogenous-price market equilibrium

Conceptual View

This specification describes a **two-agent stochastic-robust equilibrium system**:

- **Renewable producer:** acts strategically to maximize profit under deep uncertainty.
- **System regulator:** designs incentives and penalties to maintain system efficiency and reliability.
- **Shared environment:** the market-clearing mechanism acts as the mediator linking their actions.

Would you like me to next **formulate this as a dynamic game structure** — i.e., define it as a two-player Stackelberg or equilibrium model (leader-follower or simultaneous decision framework)? That would make the interaction between the renewable producer and regulator mathematically explicit.



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

Understood — let's now **tighten the model** into a fully **implementable and quantifiable specification** that retains only measurable, computationally definable elements.

All conceptual or qualitative components (e.g., "social welfare", "fairness") will be removed or rewritten using quantifiable metrics.

We'll present the specification in three layers:

1. **Common environment** (shared across both agents)
 2. **Renewable producer model**
 3. **System regulator model**
-

I. Common Environment (Shared System)

All stakeholders operate within the same 24-hour day-ahead electricity market.

1. Time Horizon

$$t = 1, 2, \dots, 24.$$

2. Global Parameters (Known Constants)

Symbol	Description	Units
\bar{G}_t	Maximum renewable generation capacity	MWh
\underline{G}_t	Minimum possible renewable generation	MWh
\bar{P}	Maximum allowable bid price	\$/MWh
P_t^u	Penalty price for under-delivery	\$/MWh
P_t^o	Penalty price for over-delivery	\$/MWh
μ_D, σ_D	Mean and standard deviation of demand D_t	MWh
μ_{b_i}, σ_{b_i}	Mean and standard deviation of bid quantity for conventional producer i	MWh
μ_{p_i}, σ_{p_i}	Mean and standard deviation of bid price for conventional producer i	\$/MWh

3. Random and Uncertain Variables

Symbol	Type	Distribution / Set	Description
D_t	Random	$\mathcal{N}(\mu_D, \sigma_D^2)$	Total electricity demand
b_{it}	Random	$\mathcal{N}(\mu_{b_i}, \sigma_{b_i}^2)$	Conventional producer i 's bid quantity
p_{it}	Random	$\mathcal{N}(\mu_{p_i}, \sigma_{p_i}^2)$	Conventional producer i 's bid price
G_t	Deep uncertainty	$[\underline{G}_t, \bar{G}_t]$	Renewable generation realization

4. Endogenous Variables (Determined by Market Clearing)

Symbol	Description	Units
P_t^c	Market clearing price	\$/MWh
q_t^r	Renewable producer's accepted quantity	MWh
q_{it}	Conventional producer i 's accepted quantity	MWh

5. Market-Clearing Mechanism (Shared Transition Function)

For each hour t , the market operator solves:

$$\begin{aligned} \min_{\{q_{it}\}, q_t^r} \quad & \sum_{i=1}^3 p_{it} q_{it} + p_t q_t^r \\ \text{s.t.} \quad & \sum_{i=1}^3 q_{it} + q_t^r = D_t, \\ & 0 \leq q_{it} \leq b_{it}, \quad 0 \leq q_t^r \leq b_t. \end{aligned}$$

The market clearing price P_t^c is the **marginal accepted offer price**, determined from optimality conditions.

II. Renewable Producer Model

1. Control Variables

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

Chosen simultaneously for all $t = 1, \dots, 24$.

2. Input (From Shared Environment)

- Random: $D_t, (b_{it}, p_{it})$
 - Uncertain: $G_t \in [\underline{G}_t, \bar{G}_t]$
 - Endogenous (feedback): P_t^c, q_t^r from market clearing
-

3. Output

- Accepted bid quantity q_t^r
 - Profit realization π_t
-

4. Hourly Profit Function

$$\pi_t(b_t, p_t, G_t, P_t^c, q_t^r) = P_t^c q_t^r - P_t^u (q_t^r - G_t)^+ + P_t^o (G_t - q_t^r)^+.$$

5. Renewable Producer Objectives

(a) Expected robust profit maximization

Maximize expected daily profit over random market variables and worst-case generation:

$$\max_{\{b_t, p_t\}} \mathbb{E}_{D_t, (b_{it}, p_{it})} \left[\min_{G_t \in [\underline{G}_t, \bar{G}_t]} \sum_{t=1}^{24} \pi_t(b_t, p_t, G_t, P_t^c, q_t^r) \right].$$

(b) Expected imbalance minimization

Minimize expected absolute deviation between scheduled and actual output:

$$\min_{\{b_t, p_t\}} \mathbb{E}_{D_t, (b_{it}, p_{it})} \left[\max_{G_t \in [\underline{G}_t, \bar{G}_t]} \sum_{t=1}^{24} |G_t - q_t^r| \right].$$

6. Constraints

$$0 \leq b_t \leq \bar{G}_t, \quad 0 \leq p_t \leq \bar{P}, \quad \forall t.$$

7. Renewable Producer Summary

Component	Symbol / Equation	Quantifiable form
Control variables	b_t, p_t	Bid quantity and price
Inputs	$D_t, (b_{it}, p_{it}), G_t$	Random or uncertain values
Transition	MarketClearing(D, bids)	Linear program
Outputs	q_t^r, P_t^c	From clearing model
Objective 1	Maximize expected robust profit	Quantitative
Objective 2	Minimize expected imbalance	Quantitative

III. System Regulator Model

1. Control Variables (Policy Levers)

For each hour t :

- P_t^u : penalty price for under-delivery (\$/MWh)
- P_t^o : penalty price for over-delivery (\$/MWh)

These are **quantifiable market parameters** directly set by the regulator.

2. Input (Shared Environment)

- Random: $D_t, (b_{it}, p_{it})$
- Renewable's bid decisions: b_t, p_t
- Endogenous market outcomes: P_t^c, q_t^r, q_{it}
- Uncertain: $G_t \in [\underline{G}_t, \bar{G}_t]$

3. Output

- Observed market outcomes (clearing prices, imbalances)
- Aggregate reliability and cost metrics

4. Quantifiable Objective Functions

(a) Expected total system cost minimization

Includes payments to all producers and imbalance penalties:

$$\min_{\{P_t^u, P_t^o\}} \mathbb{E}_{D_t, (b_{it}, p_{it})} \left[\sum_{t=1}^{24} (P_t^c D_t + P_t^u (q_t^r - G_t)^+ + P_t^o (G_t - q_t^r)^+) \right].$$

(b) Expected imbalance minimization

Reduce total expected deviation between scheduled and actual production:

$$\min_{\{P_t^u, P_t^o\}} \mathbb{E}_{D_t, (b_{it}, p_{it})} \left[\max_{G_t \in [\underline{G}_t, \bar{G}_t]} \sum_{t=1}^{24} |G_t - q_t^r| \right].$$

5. Constraints

$$0 \leq P_t^o \leq P_t^c \leq P_t^u \leq \bar{P}, \quad \forall t.$$

6. System Regulator Summary

Component	Symbol / Equation	Quantifiable form
Control variables	P_t^u, P_t^o	Penalty prices
Inputs	$D_t, (b_{it}, p_{it}), b_t, p_t, G_t$	Market data and bids
Transition	MarketClearing(D, bids)	Shared function
Outputs	P_t^c, q_t^r, q_{it}	Endogenous variables
Objective 1	Minimize expected system cost	Quantitative
Objective 2	Minimize expected imbalance	Quantitative
Constraints	$0 \leq P_t^o \leq P_t^c \leq P_t^u \leq \bar{P}$	Implementable bounds

IV. Summary: Shared System Structure

Element	Renewable Producer	Regulator	Shared Environment
Decision variables	b_t, p_t	P_t^u, P_t^o	P_t^c, q_t^r, q_{it}
Random variables	$D_t, (b_{it}, p_{it})$	Same	Same
Deep uncertainty	$G_t \in [\underline{G}_t, \bar{G}_t]$	Same	Same

Element	Renewable Producer	Regulator	Shared Environment
Coupling mechanism	Market clearing equations	Market clearing equations	Shared LP equilibrium
Objective 1	Max expected robust profit	Min expected system cost	Linked via P_t^c, q_t^r
Objective 2	Min expected imbalance	Min expected imbalance	Common reliability metric

✓ Implementability Notes

- Every quantity (bids, penalties, prices, capacities, deviations) is **numerical** and directly measurable.
- Uncertainties are either **parametrically distributed** (normal) or **bounded** (intervals).
- Market clearing is a **linear program**, computable per hour.
- Both agents' objectives are **expressed as expectations or min-max functions** over these quantifiable elements.

Would you like me to **express this final specification in mathematical programming form** (e.g., explicit bilevel optimization pair for regulator and producer), suitable for numerical implementation or simulation?



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 — now we'll consolidate everything into a **single, unified multi-agent system model** that:

1. Embeds both the **renewable producer** and **system regulator** as distinct decision-making perspectives;
2. Shares a **common, fully quantifiable environment**;
3. Allows each perspective to be **evaluated independently** (i.e., modularly), treating the other's controls as exogenous; and
4. Preserves all measurable and implementable components from the prior specifications.

◆ Unified Multi-Agent Market Model

We formalize the coupled day-ahead electricity market as a **shared-environment, two-controller system**:

$$\mathcal{M} = (\mathcal{E}, \mathcal{A}_r, \mathcal{A}_g, \mathcal{T}, \mathcal{O}_r, \mathcal{O}_g),$$

where:

- \mathcal{E} : common environment (shared parameters, random variables, endogenous states)
 - \mathcal{A}_r : renewable producer's submodel
 - \mathcal{A}_g : system regulator's submodel
 - \mathcal{T} : shared transition / equilibrium mapping (market clearing)
 - $\mathcal{O}_r, \mathcal{O}_g$: each perspective's objective functions.
-

I. Common Environment \mathcal{E}

1. Time

Discrete hourly intervals:

$$t = 1, \dots, 24.$$

2. Global Parameters

Symbol	Description	Units
\bar{G}_t	Max renewable generation	MWh
\underline{G}_t	Min renewable generation	MWh
\bar{P}	Market price cap	\$/MWh
μ_D, σ_D	Mean and st.dev. of demand	MWh
μ_{b_i}, σ_{b_i}	Mean and st.dev. of conventional bid quantities	MWh
μ_{p_i}, σ_{p_i}	Mean and st.dev. of conventional offer prices	\$/MWh
$i = 1, 2, 3$	Conventional producers	—

3. Random and Uncertain Variables

Symbol	Type	Domain / Distribution
D_t	Random	$\mathcal{N}(\mu_D, \sigma_D^2)$
b_{it}	Random	$\mathcal{N}(\mu_{b_i}, \sigma_{b_i}^2)$
p_{it}	Random	$\mathcal{N}(\mu_{p_i}, \sigma_{p_i}^2)$
G_t	Deep uncertainty	$[\underline{G}_t, \bar{G}_t]$

All agents observe or estimate these through forecasts.

4. Endogenous Variables (shared state variables)

Symbol	Description
P_t^c	Market clearing price
q_t^r	Renewable accepted quantity

Symbol	Description
q_{it}	Accepted quantity of conventional producer i

5. Shared Transition Function \mathcal{T} : Market Clearing

Given all bids and realized demand:

$$\begin{aligned} \text{MarketClearing}(D_t, \mathbf{b}_t, \mathbf{p}_t) = \arg \min_{\{q_{it}\}, q_t^r} & \left[\sum_{i=1}^3 p_{it} q_{it} + p_t q_t^r \right] \\ \text{s.t. } & \sum_{i=1}^3 q_{it} + q_t^r = D_t, \\ & 0 \leq q_{it} \leq b_{it}, \quad 0 \leq q_t^r \leq b_t. \end{aligned}$$

The marginal accepted offer determines P_t^c .

This mapping \mathcal{T} defines how **shared states** (P_t^c, q_t^r, q_{it}) evolve from all agents' controls and random variables.

II. Renewable Producer Submodel \mathcal{A}_r

1. Control Variables

$$u_t^r = (b_t, p_t), \quad 0 \leq b_t \leq \bar{G}_t, \quad 0 \leq p_t \leq \bar{P}.$$

2. Inputs

- Exogenous controls: regulator's penalties (P_t^u, P_t^o)
- Random variables: $D_t, (b_{it}, p_{it})$
- Deep uncertainty: $G_t \in [\underline{G}_t, \bar{G}_t]$

3. Transition (from environment)

$$(P_t^c, q_t^r) = \mathcal{T}(D_t, \mathbf{b}_t, \mathbf{p}_t).$$

4. Performance Metrics

(a) Profit per hour

$$\pi_t = P_t^c q_t^r - P_t^u (q_t^r - G_t)^+ + P_t^o (G_t - q_t^r)^+.$$

(b) Imbalance per hour

$$\delta_t = |G_t - q_t^r|.$$

5. Objectives

$$\mathcal{O}_r^{(1)}(u^r; u^g) = \mathbb{E}_{D_t, (b_{it}, p_{it})} \left[\min_{G_t \in [\underline{G}_t, \bar{G}_t]} \sum_{t=1}^{24} \pi_t \right], \quad (\text{expected robust profit})$$

$$\mathcal{O}_r^{(2)}(u^r; u^g) = \mathbb{E}_{D_t, (b_{it}, p_{it})} \left[\max_{G_t \in [\underline{G}_t, \bar{G}_t]} \sum_{t=1}^{24} \delta_t \right], \quad (\text{expected imbalance}).$$

Here $u^r = \{b_t, p_t\}_{t=1}^{24}$ are the renewable's controls, and $u^g = \{P_t^u, P_t^o\}_{t=1}^{24}$ are exogenous regulator controls.

6. Renewable Submodel Summary

Element	Symbol / Expression	Notes
Control variables	b_t, p_t	Renewable bids
Exogenous inputs	P_t^u, P_t^o	Regulator policies
Transition	MarketClearing(D, bids)	Shared function
Outputs	q_t^r, P_t^c	From clearing
Objective 1	$\mathcal{O}_r^{(1)}$	Max expected robust profit
Objective 2	$\mathcal{O}_r^{(2)}$	Min expected imbalance

III. System Regulator Submodel \mathcal{A}_g

1. Control Variables

$$u_t^g = (P_t^u, P_t^o), \quad 0 \leq P_t^o \leq P_t^c \leq P_t^u \leq \bar{P}.$$

2. Inputs

- Exogenous: renewable's bids (b_t, p_t)
- Random: $D_t, (b_{it}, p_{it})$
- Deep uncertainty: $G_t \in [\underline{G}_t, \bar{G}_t]$
- Endogenous: $(P_t^c, q_t^r, q_{it}) = \mathcal{T}(D_t, \mathbf{b}_t, \mathbf{p}_t)$

3. Performance Metrics

(a) System cost per hour

$$C_t = P_t^c D_t + P_t^u (q_t^r - G_t)^+ + P_t^o (G_t - q_t^r)^+.$$

(b) System imbalance per hour

$$\Delta_t = |G_t - q_t^r|.$$

4. Objectives

$$\mathcal{O}_g^{(1)}(u^g; u^r) = \mathbb{E}_{D_t, (b_{it}, p_{it})} \left[\sum_{t=1}^{24} C_t \right], \quad (\text{expected total system cost}),$$

$$\mathcal{O}_g^{(2)}(u^g; u^r) = \mathbb{E}_{D_t, (b_{it}, p_{it})} \left[\max_{G_t \in [G_t, \bar{G}_t]} \sum_{t=1}^{24} \Delta_t \right], \quad (\text{expected imbalance}).$$

5. Regulator Submodel Summary

Element	Symbol / Expression	Notes
Control variables	P_t^u, P_t^o	Penalty settings
Exogenous inputs	b_t, p_t	Renewable bids
Transition	MarketClearing(D, bids)	Shared function
Outputs	P_t^c, q_t^r, q_{it}	From clearing
Objective 1	$\mathcal{O}_g^{(1)}$	Min expected system cost
Objective 2	$\mathcal{O}_g^{(2)}$	Min expected imbalance

IV. Unified Model Relationships

1. Coupled System Mapping

For each hour t :

$$(P_t^c, q_t^r, q_{it}) = \mathcal{T}(D_t, (b_{it}, p_{it})_{i=1}^3, b_t, p_t),$$

which serves as the **state-transition function** shared by both agents.

2. Coupled Objectives Summary

Perspective	Objective 1	Objective 2
Renewable producer	$\max \mathcal{O}_r^{(1)}$	$\min \mathcal{O}_r^{(2)}$
System regulator	$\min \mathcal{O}_g^{(1)}$	$\min \mathcal{O}_g^{(2)}$

Each agent treats the other's controls as **given inputs** during its own optimization.

3. Shared State Dynamics

At each hour t :

$$\begin{aligned} \text{State vector: } s_t &= (D_t, G_t, b_t, p_t, \{b_{it}, p_{it}\}, P_t^u, P_t^o), \\ \text{Transition: } s_{t+1} &= f(s_t, \mathcal{T}(s_t)), \end{aligned}$$

where $f(\cdot)$ aggregates hourly market outcomes into daily totals (e.g., cost, profit, imbalance).

V. Compact Form of Unified Model

$$\left\{ \begin{array}{ll} \textbf{Environment:} & (P_t^c, q_t^r, q_{it}) = \mathcal{T}(D_t, (b_{it}, p_{it}), b_t, p_t), \\ \textbf{Renewable producer:} & \max_{b_t, p_t} \mathcal{O}_r^{(1)}(u^r; u^g), \\ \textbf{System regulator:} & \min_{b_t, p_t} \mathcal{O}_r^{(2)}(u^r; u^g), \\ & \min_{P_t^u, P_t^o} \mathcal{O}_g^{(1)}(u^g; u^r), \\ & \min_{P_t^u, P_t^o} \mathcal{O}_g^{(2)}(u^g; u^r). \end{array} \right.$$

VI. Modular Evaluation Capability

- **Independent evaluation:**
Each perspective's optimization can be solved holding the other's control vector fixed.
- **Shared environment simulation:**
Common random draws for $D_t, (b_{it}, p_{it}), G_t$ ensure consistent scenario evaluation.
- **Coupling through equilibrium:**
The transition function \mathcal{T} ensures consistency of market price and accepted quantities across both models.

✓ Key Properties

Property	Description
Internal consistency	All objectives depend on shared, quantifiable states P_t^c, q_t^r from the same clearing mechanism.
Implementability	All variables measurable; all relationships computable via LP and expectations.
Modularity	Each agent's optimization treats the other's controls as fixed inputs.
Composability	Both submodels embedded in one unified system through the shared transition \mathcal{T} .

Final Remarks

This unified formulation provides a consistent, implementable mathematical framework that integrates:

- **Renewable producer control and performance evaluation,**
- **Regulator policy control and performance evaluation,** and
- **A shared stochastic-robust market environment** governing their interaction.

Each perspective can be solved, simulated, or optimized modularly, yet both operate coherently within the same quantifiable system state and transition dynamics.



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.

Below is a **modular, object-oriented Python implementation** of the unified model.

It includes:

- A shared **MarketEnvironment** class for global parameters and the market-clearing mechanism,
- Two perspective classes: **RenewableProducer** and **SystemRegulator**,
- A unified interface function **simulate_market()** that accepts dictionaries of control variables and uncertain parameter values, simulates the environment, and returns all objective values.

```
python

import numpy as np

# =====
# Shared Environment Definition
# =====

class MarketEnvironment:
    """
    Shared environment for the day-ahead electricity market.
    Includes random variables, uncertain parameters, and market clearing.
    """

    def __init__(self, params):
        """
        params: dictionary of global constants
        {
            'T': 24,
            'mu_D': float, 'sigma_D': float,
            'mu_b': [float]*3, 'sigma_b': [float]*3,
            'mu_p': [float]*3, 'sigma_p': [float]*3,
            'G_bounds': [(float, float)]*24,
            'price_cap': float
        }
        """
        self.params = params
        self.T = params.get("T", 24)

    def sample_random_variables(self):
        """
        Sample demand and conventional producer bids.
        """
        mu_D, sigma_D = self.params["mu_D"], self.params["sigma_D"]
        mu_b, sigma_b = self.params["mu_b"], self.params["sigma_b"]
        mu_p, sigma_p = self.params["mu_p"], self.params["sigma_p"]

        D = np.random.normal(mu_D, sigma_D, self.T)
        b_i = [np.random.normal(mu_b[i], sigma_b[i], self.T) for i in range(3)]
        p_i = [np.random.normal(mu_p[i], sigma_p[i], self.T) for i in range(3)]

        return D, b_i, p_i

    def market_clearing(self, D_t, bids_conventional, bids_renewable):
        """
        Simulate market clearing for a single hour.
        - D_t: realized demand
        - bids_conventional: list of (b_i, p_i)
        - bids_renewable: (b_r, p_r)
        Returns: clearing price P_c, accepted quantities (q_i, q_r)
        """

```

```

offers = []
for i, (b_i, p_i) in enumerate(bids_conventional):
    offers.append((p_i, b_i, f"conv_{i}"))
offers.append((bids_renewable[1], bids_renewable[0], "renewable"))

# Sort offers by price (merit order)
offers.sort(key=lambda x: x[0])

remaining_demand = D_t
q = {}
P_c = 0.0
for price, quantity, name in offers:
    accepted = min(quantity, remaining_demand)
    q[name] = accepted
    remaining_demand -= accepted
    if remaining_demand <= 1e-6:
        P_c = price
        break

# Fill zeroes for unaccepted
for _, _, name in offers:
    q.setdefault(name, 0.0)

q_i = [q.get(f"conv_{i}", 0.0) for i in range(3)]
q_r = q.get("renewable", 0.0)
return P_c, q_i, q_r

# =====
# Renewable Producer Perspective
# =====

class RenewableProducer:
    def __init__(self, env: MarketEnvironment, controls):
        """
        controls: dictionary with keys:
            'b': array-like of bid quantities [24],
            'p': array-like of offer prices [24]
        """
        self.env = env
        self.b = np.array(controls["b"])
        self.p = np.array(controls["p"])

    def profit_per_hour(self, P_c, q_r, G_t, P_u, P_o):
        """Compute hourly profit given parameters."""
        return P_c * q_r - P_u * max(0, q_r - G_t) + P_o * max(0, G_t - q_r)

    def imbalance_per_hour(self, G_t, q_r):
        """Compute absolute imbalance."""
        return abs(G_t - q_r)

    def evaluate_objectives(self, env_realizations, penalties):
        """
        Evaluate expected profit and imbalance.
        env_realizations: (D, b_i, p_i) from environment
        penalties: dict with P_u, P_o arrays
        Returns: (expected_robust_profit, expected_imbalance)
        """
        D, b_i, p_i = env_realizations
        T = self.env.T
        G_bounds = self.env.params["G_bounds"]
        P_u, P_o = np.array(penalties["P_u"]), np.array(penalties["P_o"])

        exp_profit, exp_imbalance = 0.0, 0.0

        for t in range(T):
            # Market clearing
            bids_conventional = [(b_i[j][t], p_i[j][t]) for j in range(3)]
            bids_renewable = (self.b[t], self.p[t])
            P_c, _, q_r = self.env.market_clearing(D[t], bids_conventional, bids_renewable)

            # Worst-case renewable generation (robust)
            G_low, G_high = G_bounds[t]
            # For worst-case profit: pick G that minimizes profit
            profit_low = self.profit_per_hour(P_c, q_r, G_low, P_u[t], P_o[t])

```

```

profit_high = self.profit_per_hour(P_c, q_r, G_high, P_u[t], P_o[t])
worst_profit = min(profit_low, profit_high)

# For imbalance: pick G that maximizes imbalance
imbalance_low = self.imbalance_per_hour(G_low, q_r)
imbalance_high = self.imbalance_per_hour(G_high, q_r)
worst_imbalance = max(imbalance_low, imbalance_high)

exp_profit += worst_profit
exp_imbalance += worst_imbalance

return exp_profit, exp_imbalance

# =====
# System Regulator Perspective
# =====

class SystemRegulator:
    def __init__(self, env: MarketEnvironment, controls):
        """
        controls: dictionary with keys:
            'P_u': array-like of under-delivery penalty prices [24],
            'P_o': array-like of over-delivery penalty prices [24]
        """
        self.env = env
        self.P_u = np.array(controls["P_u"])
        self.P_o = np.array(controls["P_o"])

    def system_cost_per_hour(self, P_c, D_t, q_r, G_t, P_u, P_o):
        """Compute total system cost per hour."""
        return P_c * D_t + P_u * max(0, q_r - G_t) + P_o * max(0, G_t - q_r)

    def evaluate_objectives(self, env_realizations, renewable_controls):
        """
        Evaluate expected total system cost and imbalance.
        env_realizations: (D, b_i, p_i)
        renewable_controls: dict with 'b', 'p'
        Returns: (expected_system_cost, expected_imbalance)
        """
        D, b_i, p_i = env_realizations
        T = self.env.T
        b_r, p_r = np.array(renewable_controls["b"]), np.array(renewable_controls["p"])
        G_bounds = self.env.params["G_bounds"]

        exp_cost, exp_imbalance = 0.0, 0.0

        for t in range(T):
            bids_conventional = [(b_i[j][t], p_i[j][t]) for j in range(3)]
            bids_renewable = (b_r[t], p_r[t])
            P_c, _, q_r = self.env.market_clearing(D[t], bids_conventional, bids_renewable)

            G_low, G_high = G_bounds[t]
            cost_low = self.system_cost_per_hour(P_c, D[t], q_r, G_low, self.P_u[t],
self.P_o[t])
            cost_high = self.system_cost_per_hour(P_c, D[t], q_r, G_high, self.P_u[t],
self.P_o[t])
            worst_cost = max(cost_low, cost_high)

            imbalance_low = abs(G_low - q_r)
            imbalance_high = abs(G_high - q_r)
            worst_imbalance = max(imbalance_low, imbalance_high)

            exp_cost += worst_cost
            exp_imbalance += worst_imbalance

        return exp_cost, exp_imbalance

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

def simulate_market(control_vars, uncertain_params):
    """
    """

```

Interface function to simulate unified model dynamics.

Inputs:

```
control_vars: dict with sub-dictionaries:
{
    'renewable': {'b': [...], 'p': [...]},
    'regulator': {'P_u': [...], 'P_o': [...]}
}
uncertain_params: dict with random & uncertain parameters:
{
    'mu_D', 'sigma_D',
    'mu_b', 'sigma_b',
    'mu_p', 'sigma_p',
    'G_bounds', 'price_cap'
}
```

Returns:

```
results: dict of objective values:
{
    'renewable_profit': float,
    'renewable_imbalance': float,
    'system_cost': float,
    'system_imbalance': float
}
"""

```

1. Initialize shared environment

```
params = uncertain_params.copy()
params["T"] = 24
env = MarketEnvironment(params)
```

2. Sample random variables

```
D, b_i, p_i = env.sample_random_variables()
```

3. Instantiate stakeholders

```
renewable = RenewableProducer(env, control_vars["renewable"])
regulator = SystemRegulator(env, control_vars["regulator"])
```

4. Evaluate objectives

```
env_realizations = (D, b_i, p_i)
penalties = control_vars["regulator"]
```

```
profit, imbalance_r = renewable.evaluate_objectives(env_realizations, penalties)
cost, imbalance_sys = regulator.evaluate_objectives(env_realizations,
```

```
control_vars["renewable"])
```

5. Return results

```
return {
    "renewable_profit": profit,
    "renewable_imbalance": imbalance_r,
    "system_cost": cost,
    "system_imbalance": imbalance_sys,
}
```

```
# =====
# Example Usage (not executed)
# =====
# results = simulate_market(control_vars, uncertain_params)
# print(results)
```

✓ Key Features of This Implementation

Feature	Description
Modular class design	Each perspective (RenewableProducer , SystemRegulator) encapsulates its own control variables, objectives, and evaluation methods.
Shared environment	The MarketEnvironment class provides a consistent stochastic context and a common market-clearing mechanism.

Feature	Description
Robust-stochastic simulation	Uses random draws for demand and competitors' bids, and worst-case (min-max) logic for deep uncertainty in renewable generation.
Unified interface	simulate_market() ties all components together, returning both agents' objective function values from a single call.
Fully quantifiable	Every model element (controls, states, objectives) is numeric and directly computable.

This structure can be readily extended for scenario analysis, optimization loops, or game-theoretic simulations — while preserving independent, modular evaluation for each stakeholder within the shared environment.



`mu_D, sigma_D, mu_b, sigma_b, mu_p, sigma_p` are constants instead of uncertain parameters, so the implementation should not take them as inputs. Please refine the implementation.

Got it — we'll refine the implementation so that:

- **Statistical parameters** (`mu_D, sigma_D, mu_b, sigma_b, mu_p, sigma_p`) are **internal constants** of the environment, not inputs to the simulation function.
- The `simulate_market()` interface now only requires:
 - control variables (renewable and regulator), and
 - uncertain parameters (`G_bounds, price_cap`).

Everything else is treated as fixed constants within the environment setup.

Here's the refined, fully self-contained implementation

```
python

import numpy as np

# =====
# Shared Environment Definition
# =====

class MarketEnvironment:
    """
    Shared environment for the day-ahead electricity market.
    Includes fixed constants, random variables, and market clearing.
    """

    def __init__(self, uncertain_params):
        """
        uncertain_params: dictionary of uncertain parameters:
        {
            'G_bounds': [(float, float)]*24,
            'price_cap': float
        }
        """
        # Fixed constants (global market settings)
        self.T = 24
        self.mu_D, self.sigma_D = 1000.0, 100.0      # demand (MWh)
        self.mu_b = [300.0, 250.0, 200.0]          # bid quantities (MWh)
        self.sigma_b = [30.0, 25.0, 20.0]
```

```

self.mu_p = [40.0, 45.0, 50.0]                      # bid prices ($/MWh)
self.sigma_p = [4.0, 4.5, 5.0]

# Uncertain parameters
self.params = uncertain_params
self.G_bounds = uncertain_params["G_bounds"]
self.price_cap = uncertain_params["price_cap"]

def sample_random_variables(self):
    """Sample demand and conventional producer bids (random, not uncertain)."""
    D = np.random.normal(self.mu_D, self.sigma_D, self.T)
    b_i = [np.random.normal(self.mu_b[i], self.sigma_b[i], self.T) for i in range(3)]
    p_i = [np.random.normal(self.mu_p[i], self.sigma_p[i], self.T) for i in range(3)]
    return D, b_i, p_i

def market_clearing(self, D_t, bids_conventional, bids_renewable):
    """
    Simulate hourly market clearing.
    - D_t: realized demand
    - bids_conventional: list of (b_i, p_i)
    - bids_renewable: (b_r, p_r)
    Returns: (P_c, q_i, q_r)
    """
    offers = []
    for i, (b_i, p_i) in enumerate(bids_conventional):
        offers.append((p_i, b_i, f"conv_{i}"))
    offers.append((bids_renewable[1], bids_renewable[0], "renewable"))

    # Sort offers by ascending price (merit order)
    offers.sort(key=lambda x: x[0])

    remaining_demand = D_t
    q = {}
    P_c = 0.0
    for price, quantity, name in offers:
        accepted = min(quantity, remaining_demand)
        q[name] = accepted
        remaining_demand -= accepted
        if remaining_demand <= 1e-6:
            P_c = price
            break

    # Fill zero quantities for unaccepted offers
    for _, _, name in offers:
        q.setdefault(name, 0.0)

    q_i = [q.get(f"conv_{i}", 0.0) for i in range(3)]
    q_r = q.get("renewable", 0.0)
    return P_c, q_i, q_r

# =====
# Renewable Producer Perspective
# =====

class RenewableProducer:
    def __init__(self, env: MarketEnvironment, controls):
        """
        controls: dictionary with keys:
        'b': array-like of bid quantities [24],
        'p': array-like of offer prices [24]
        """
        self.env = env
        self.b = np.array(controls["b"])
        self.p = np.array(controls["p"])

    def profit_per_hour(self, P_c, q_r, G_t, P_u, P_o):
        """Compute hourly profit."""
        return P_c * q_r - P_u * max(0, q_r - G_t) + P_o * max(0, G_t - q_r)

    def imbalance_per_hour(self, G_t, q_r):
        """Compute absolute imbalance."""
        return abs(G_t - q_r)

    def evaluate_objectives(self, env_realizations, penalties):

```

```

"""
Evaluate expected robust profit and expected imbalance.
env_realizations: (D, b_i, p_i)
penalties: {'P_u': [...], 'P_o': [...]}

D, b_i, p_i = env_realizations
T = self.env.T
G_bounds = self.env.G_bounds
P_u, P_o = np.array(penalties["P_u"]), np.array(penalties["P_o"])

exp_profit, exp_imbalance = 0.0, 0.0

for t in range(T):
    # Market clearing
    bids_conventional = [(b_i[j][t], p_i[j][t]) for j in range(3)]
    bids_renewable = (self.b[t], self.p[t])
    P_c, _, q_r = self.env.market_clearing(D[t], bids_conventional, bids_renewable)

    # Worst-case renewable generation (deep uncertainty)
    G_low, G_high = G_bounds[t]
    profit_low = self.profit_per_hour(P_c, q_r, G_low, P_u[t], P_o[t])
    profit_high = self.profit_per_hour(P_c, q_r, G_high, P_u[t], P_o[t])
    worst_profit = min(profit_low, profit_high)

    imbalance_low = self.imbalance_per_hour(G_low, q_r)
    imbalance_high = self.imbalance_per_hour(G_high, q_r)
    worst_imbalance = max(imbalance_low, imbalance_high)

    exp_profit += worst_profit
    exp_imbalance += worst_imbalance

return exp_profit, exp_imbalance

# =====
# System Regulator Perspective
# =====

class SystemRegulator:
    def __init__(self, env: MarketEnvironment, controls):
        """
        controls: dictionary with keys:
            'P_u': array-like [24],
            'P_o': array-like [24]
        """
        self.env = env
        self.P_u = np.array(controls["P_u"])
        self.P_o = np.array(controls["P_o"])

    def system_cost_per_hour(self, P_c, D_t, q_r, G_t, P_u, P_o):
        """Compute total system cost for one hour."""
        return P_c * D_t + P_u * max(0, q_r - G_t) + P_o * max(0, G_t - q_r)

    def evaluate_objectives(self, env_realizations, renewable_controls):
        """
        Evaluate expected system cost and imbalance.
        env_realizations: (D, b_i, p_i)
        renewable_controls: {'b': [...], 'p': [...]}

        D, b_i, p_i = env_realizations
        T = self.env.T
        b_r, p_r = np.array(renewable_controls["b"]), np.array(renewable_controls["p"])
        G_bounds = self.env.G_bounds

        exp_cost, exp_imbalance = 0.0, 0.0

        for t in range(T):
            bids_conventional = [(b_i[j][t], p_i[j][t]) for j in range(3)]
            bids_renewable = (b_r[t], p_r[t])
            P_c, _, q_r = self.env.market_clearing(D[t], bids_conventional, bids_renewable)

            G_low, G_high = G_bounds[t]
            cost_low = self.system_cost_per_hour(P_c, D[t], q_r, G_low, self.P_u[t],
                                                self.P_o[t])
            cost_high = self.system_cost_per_hour(P_c, D[t], q_r, G_high, self.P_u[t],
                                                self.P_o[t])
            exp_cost += (cost_low + cost_high) / 2
            exp_imbalance += (G_low - G_high) / 2
        """

```

```

self.P_o[t])
    worst_cost = max(cost_low, cost_high)

    imbalance_low = abs(G_low - q_r)
    imbalance_high = abs(G_high - q_r)
    worst_imbalance = max(imbalance_low, imbalance_high)

    exp_cost += worst_cost
    exp_imbalance += worst_imbalance

    return exp_cost, exp_imbalance

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

def simulate_market(control_vars, uncertain_params):
    """
    Simulate unified market dynamics and return each perspective's objective values.

    Parameters
    -----
    control_vars : dict
        {
            'renewable': {'b': [...], 'p': [...]},
            'regulator': {'P_u': [...], 'P_o': [...]}
        }
    uncertain_params : dict
        {
            'G_bounds': [(low, high)] * 24,
            'price_cap': float
        }

    Returns
    -----
    results : dict
        {
            'renewable_profit': float,
            'renewable_imbalance': float,
            'system_cost': float,
            'system_imbalance': float
        }
    """

# 1. Initialize shared environment with fixed constants and uncertain params
env = MarketEnvironment(uncertain_params)

# 2. Sample random demand and conventional bids
D, b_i, p_i = env.sample_random_variables()
env_realizations = (D, b_i, p_i)

# 3. Instantiate both perspectives
renewable = RenewableProducer(env, control_vars["renewable"])
regulator = SystemRegulator(env, control_vars["regulator"])

# 4. Evaluate each perspective's objectives
penalties = control_vars["regulator"]
profit, imbalance_r = renewable.evaluate_objectives(env_realizations, penalties)
cost, imbalance_sys = regulator.evaluate_objectives(env_realizations,
control_vars["renewable"])

# 5. Return all objective values
return {
    "renewable_profit": profit,
    "renewable_imbalance": imbalance_r,
    "system_cost": cost,
    "system_imbalance": imbalance_sys,
}

# =====
# Example (not executed)
# =====
# control_vars = {
#     'renewable': {'b': [100]*24, 'p': [45]*24},
#

```

```
#     'regulator': {'P_u': [60]*24, 'P_o': [30]*24}
# }
# uncertain_params = {
#     'G_bounds': [(80, 120)]*24,
#     'price_cap': 100.0
# }
# results = simulate_market(control_vars, uncertain_params)
# print(results)
```

✓ Implementation Summary

Feature	Description
Fixed constants	<code>mu_D, sigma_D, mu_b, sigma_b, mu_p, sigma_p</code> are built-in inside <code>MarketEnvironment</code> .
Inputs to simulation	Only <code>control_vars</code> (decisions) and <code>uncertain_params (G_bounds, price_cap)</code> .
Outputs	Quantitative objective values for both perspectives.
Structure	Modular classes: <code>MarketEnvironment, RenewableProducer, SystemRegulator</code> .
Interface	<code>simulate_market()</code> integrates all components into a single callable entry point.

This final version ensures internal consistency, full quantifiability, and modular evaluation capability for both perspectives — all within one coherent simulation framework.



Both `P_u` and `P_o` are penalty coefficients in the function `profit_per_hour`. Please refine that.

Excellent catch — you're absolutely right.

In the previous version, P_u and P_o were treated as prices, but in the model specification they are **penalty coefficients** that scale deviations (i.e., over- and under-delivery penalties) rather than additive price replacements.

We'll refine the implementation so that:

- P_u and P_o act as **penalty coefficients**, applied to deviations relative to the clearing price P_c .
- The profit function becomes:

$$\pi_t = P_c q_r - P_c P_u (q_r - G_t)^+ - P_c P_o (G_t - q_r)^+$$

where P_u and P_o are **dimensionless fractions** (e.g. 0.2 for 20% penalty).

- The regulator's system cost function also uses these coefficients consistently:

$$C_t = P_c D_t + P_c P_u (q_r - G_t)^+ + P_c P_o (G_t - q_r)^+$$

Here's the refined, fully consistent, **quantifiable and modular** implementation

python

```
import numpy as np

# =====
# Shared Environment Definition
```

```

# =====
# class MarketEnvironment:
#     """
#         Shared environment for the day-ahead electricity market.
#         Includes random variables, uncertain parameters, and market clearing.
#     """

#     def __init__(self, uncertain_params):
#         """
#             uncertain_params: dictionary of uncertain parameters
#             {
#                 'G_bounds': [(float, float)]*24,
#                 'price_cap': float
#             }
#         """
#         # Fixed constants (global market settings)
#         self.T = 24
#         self.mu_D, self.sigma_D = 1000.0, 100.0      # demand (MWh)
#         self.mu_b = [300.0, 250.0, 200.0]           # bid quantities (MWh)
#         self.sigma_b = [30.0, 25.0, 20.0]
#         self.mu_p = [40.0, 45.0, 50.0]                # bid prices ($/MWh)
#         self.sigma_p = [4.0, 4.5, 5.0]

#         # Uncertain parameters
#         self.G_bounds = uncertain_params["G_bounds"]
#         self.price_cap = uncertain_params["price_cap"]

#     def sample_random_variables(self):
#         """
#             Sample demand and conventional producer bids.
#         """
#         D = np.random.normal(self.mu_D, self.sigma_D, self.T)
#         b_i = [np.random.normal(self.mu_b[i], self.sigma_b[i], self.T) for i in range(3)]
#         p_i = [np.random.normal(self.mu_p[i], self.sigma_p[i], self.T) for i in range(3)]
#         return D, b_i, p_i

#     def market_clearing(self, D_t, bids_conventional, bids_renewable):
#         """
#             Simulate hourly market clearing.
#             Returns: clearing price P_c, accepted quantities q_i (list), q_r (float)
#         """
#         offers = []
#         for i, (b_i, p_i) in enumerate(bids_conventional):
#             offers.append((p_i, b_i, f"conv_{i}"))
#         offers.append((bids_renewable[1], bids_renewable[0], "renewable"))

#         # Sort by ascending price (merit order)
#         offers.sort(key=lambda x: x[0])

#         remaining_demand = D_t
#         q = {}
#         P_c = 0.0
#         for price, quantity, name in offers:
#             accepted = min(quantity, remaining_demand)
#             q[name] = accepted
#             remaining_demand -= accepted
#             if remaining_demand <= 1e-6:
#                 P_c = price
#                 break

#         # Fill zeroes for unaccepted offers
#         for _, _, name in offers:
#             q.setdefault(name, 0.0)

#         q_i = [q.get(f"conv_{i}", 0.0) for i in range(3)]
#         q_r = q.get("renewable", 0.0)
#         return P_c, q_i, q_r

#     # =====
#     # Renewable Producer Perspective
#     # =====

#     class RenewableProducer:
#         def __init__(self, env: MarketEnvironment, controls):
#             """
#         """

```

```

controls: {'b': [24], 'p': [24]}
"""
self.env = env
self.b = np.array(controls["b"])
self.p = np.array(controls["p"])

def profit_per_hour(self, P_c, q_r, G_t, P_u, P_o):
    """
    Hourly profit with penalty coefficients P_u, P_o (fractions of clearing price).
    """
    penalty_under = P_c * P_u * max(0, q_r - G_t)
    penalty_over = P_c * P_o * max(0, G_t - q_r)
    return P_c * q_r - penalty_under - penalty_over

def imbalance_per_hour(self, G_t, q_r):
    """
    Absolute deviation between scheduled and realized generation.
    """
    return abs(G_t - q_r)

def evaluate_objectives(self, env_realizations, penalties):
    """
    Compute expected robust profit and expected imbalance.
    env_realizations: (D, b_i, p_i)
    penalties: {'P_u': [...], 'P_o': [...]}

    D, b_i, p_i = env_realizations
    T = self.env.T
    G_bounds = self.env.G_bounds
    P_u, P_o = np.array(penalties["P_u"]), np.array(penalties["P_o"])

    exp_profit, exp_imbalance = 0.0, 0.0

    for t in range(T):
        bids_conventional = [(b_i[j][t], p_i[j][t]) for j in range(3)]
        bids_renewable = (self.b[t], self.p[t])
        P_c, _, q_r = self.env.market_clearing(D[t], bids_conventional, bids_renewable)

        # Worst-case renewable generation
        G_low, G_high = G_bounds[t]
        profit_low = self.profit_per_hour(P_c, q_r, G_low, P_u[t], P_o[t])
        profit_high = self.profit_per_hour(P_c, q_r, G_high, P_u[t], P_o[t])
        worst_profit = min(profit_low, profit_high)

        imbalance_low = self.imbalance_per_hour(G_low, q_r)
        imbalance_high = self.imbalance_per_hour(G_high, q_r)
        worst_imbalance = max(imbalance_low, imbalance_high)

        exp_profit += worst_profit
        exp_imbalance += worst_imbalance

    return exp_profit, exp_imbalance

# =====
# System Regulator Perspective
# =====

class SystemRegulator:
    def __init__(self, env: MarketEnvironment, controls):
        """
        controls: {'P_u': [24], 'P_o': [24]} # penalty coefficients
        """
        self.env = env
        self.P_u = np.array(controls["P_u"])
        self.P_o = np.array(controls["P_o"])

    def system_cost_per_hour(self, P_c, D_t, q_r, G_t, P_u, P_o):
        """
        Hourly total system cost including penalty coefficients.
        """
        penalty_under = P_c * P_u * max(0, q_r - G_t)
        penalty_over = P_c * P_o * max(0, G_t - q_r)
        return P_c * D_t + penalty_under + penalty_over

    def evaluate_objectives(self, env_realizations, renewable_controls):
        """
        """

```

```

Compute expected total system cost and expected imbalance.
"""
D, b_i, p_i = env_realizations
T = self.env.T
b_r, p_r = np.array(renewable_controls["b"]), np.array(renewable_controls["p"])
G_bounds = self.env.G_bounds

exp_cost, exp_imbalance = 0.0, 0.0

for t in range(T):
    bids_conventional = [(b_i[j][t], p_i[j][t]) for j in range(3)]
    bids_renewable = (b_r[t], p_r[t])
    P_c, _, q_r = self.env.market_clearing(D[t], bids_conventional, bids_renewable)

    G_low, G_high = G_bounds[t]
    cost_low = self.system_cost_per_hour(P_c, D[t], q_r, G_low, self.P_u[t],
self.P_o[t])
    cost_high = self.system_cost_per_hour(P_c, D[t], q_r, G_high, self.P_u[t],
self.P_o[t])
    worst_cost = max(cost_low, cost_high)

    imbalance_low = abs(G_low - q_r)
    imbalance_high = abs(G_high - q_r)
    worst_imbalance = max(imbalance_low, imbalance_high)

    exp_cost += worst_cost
    exp_imbalance += worst_imbalance

return exp_cost, exp_imbalance

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

def simulate_market(control_vars, uncertain_params):
    """
    Simulate unified market dynamics and compute all objective values.

    Parameters
    -----
    control_vars : dict
        {
            'renewable': {'b': [...], 'p': [...]},
            'regulator': {'P_u': [...], 'P_o': [...]}
        }
    uncertain_params : dict
        {
            'G_bounds': [(low, high)] * 24,
            'price_cap': float
        }

    Returns
    -----
    results : dict
        {
            'renewable_profit': float,
            'renewable_imbalance': float,
            'system_cost': float,
            'system_imbalance': float
        }
    """

    # Initialize environment (fixed constants + uncertain inputs)
    env = MarketEnvironment(uncertain_params)

    # Sample random demand and conventional producer bids
    D, b_i, p_i = env.sample_random_variables()
    env_realizations = (D, b_i, p_i)

    # Instantiate perspectives
    renewable = RenewableProducer(env, control_vars["renewable"])
    regulator = SystemRegulator(env, control_vars["regulator"])

    # Evaluate each perspective
    penalties = control_vars["regulator"]

```

```

profit, imbalance_r = renewable.evaluate_objectives(env_realizations, penalties)
cost, imbalance_sys = regulator.evaluate_objectives(env_realizations,
control_vars["renewable"])

return {
    "renewable_profit": profit,
    "renewable_imbalance": imbalance_r,
    "system_cost": cost,
    "system_imbalance": imbalance_sys,
}

# =====
# Example (not executed)
# =====
# control_vars = {
#     'renewable': {'b': [100]*24, 'p': [45]*24},
#     'regulator': {'P_u': [0.3]*24, 'P_o': [0.1]*24} # penalty coefficients
# }
# uncertain_params = {
#     'G_bounds': [(80, 120)]*24,
#     'price_cap': 100.0
# }
# results = simulate_market(control_vars, uncertain_params)
# print(results)

```

✓ Summary of Refinements

Aspect	Previous Version	Refined Version
Penalty interpretation	P_u, P_o as additive prices	P_u, P_o as <i>coefficients</i> applied to clearing price P_c
Profit formula	$P_c q_r - P_u(q_r - G)^+ + P_o(G - q_r)^+$	$P_c q_r - P_c P_u(q_r - G)^+ - P_c P_o(G - q_r)^+$
System cost formula	$P_c D_t + P_u(\dots) + P_o(\dots)$	$P_c D_t + P_c P_u(\dots) + P_c P_o(\dots)$
Interpretation	Penalties in \$/MWh	Penalty <i>fractions</i> (dimensionless coefficients)
Output structure	Unchanged	Same dictionary interface; all numeric objectives

This refined implementation is now **fully consistent** with the model specification — all components quantifiable, internally coherent, and directly executable in simulation or optimization workflows.