Final Workflow Explanation for Load Allocation Optimization

This document summarizes our process for building a load allocation optimization model using synthetic data. It outlines the key steps, the rationale behind each decision, and how we designed the system to be easily adjusted when real data become available.

---

1. Data Ingestion and Winsorization

A. Data Sources

- **Synthetic Data Files:**
  We use three CSV files (e.g., df_fixed_winsorized.csv and df_quantile_winsorized.csv).
- **Winsorization Methods:**
  - **Fixed Threshold Winsorization:**
    Caps values at predetermined limits (e.g., Plant Load capped at 100 MWh and SPOT price capped at 200 EUR/MWh).
  - **Quantile-Based Winsorization:**
    Uses the 1st and 99th percentiles to cap extreme values.
- **Pros and Cons:**
  - *Quantile-Based:*
    - **Pros:** Automatically adapts to changes in data distribution.
    - **Cons:** May not match real-world benchmarks as the synthetic data are not representative.
  - *Fixed Threshold:*
    - **Pros:** Provides consistent limits based on known or assumed operational ranges based on the data provided.
    - **Cons:** Less flexible if the data distribution shifts over time.

B. Aggregation of Weekly Data

- **Purpose:**
  To aggregate hourly data into a weekly profile, ensuring that the optimization uses mean values per time slot.
- **Method:**
  - Convert the Timestamp column to datetime.
  - Filter the data for the target week.
  - Resample to hourly frequency, aggregating only numeric columns.

This aggregated data (df_week) is used as input for the optimization model.

---

2. Optimization Problem Formulation

A. Decision Variable and Bounds

- **Decision Variable:**
  A vector x∈Rnx \in \mathbb{R}^{n}x∈Rn representing the allocated load (in MWh) for each of nnn hourly time slots.
- **Hourly Bounds:**
  - **Minimum Hourly Load:** min_hourly=0.00001×tpl\text{min_hourly} = 0.00001 \times \text{tpl}min_hourly=0.00001×tpl
  - **Maximum Hourly Load:** max_hourly=0.1×tpl\text{max_hourly} = 0.1 \times \text{tpl}max_hourly=0.1×tpl

Note: These bounds are set based on the given data. These parameters can be tuned accordingly to your real data.

B. Total Load Constraint

- **Requirement:**
  The sum of hourly loads must equal the Total Plant Load (TPL) for the week, with a tolerance of ±5%.
- **Constraint:** (1−tol)×tpl≤∑i=1nxi≤(1+tol)×tpl(1 - tol) \times \text{tpl} \leq \sum_{i=1}^{n} x_i \leq (1 + tol) \times \text{tpl}(1−tol)×tpl≤i=1∑nxi≤(1+tol)×tpl where tol=0.05tol = 0.05tol=0.05.

C. Objective Function

- **Cost Minimization:**
  The objective is to minimize the total cost, defined as:
  Cost=∑i=1n(SPOT price at slot i)×xi\text{Cost} = \sum_{i=1}^{n} (\text{SPOT price at slot } i) \times x_iCost=i=1∑n(SPOT price at slot i)×xi

D. Forced Constraint / Soft Constraint for High SPOT Prices

- **Challenge:**
  In real operations, if SPOT prices are very high, it may be desirable to allocate load according to the Hedged Base Band (HBB) volume.
- **Forced Constraint Issue:**
  The given data showed that a hard forced constraint (forcing xi=min(HBB,max_hourly)x_i = \min(\text{HBB}, \text{max_hourly})xi =min(HBB,max_hourly)) was too aggressive—triggering in almost every time slot and leading to infeasibility.
- **Solution: Soft Constraint Approach**
  Instead of hard-forcing, we add a penalty term to the objective when the SPOT price exceeds a threshold. The threshold is computed as:
  threshold=1.5×HBP×scaling_factor\text{threshold} = 1.5 \times \text{HBP} \times \text{scaling_factor}threshold=1.5×HBP×scaling_factor where HBP (Hedged Base Band Price) is computed as HBC/HBB\text{HBC} / \text{HBB}HBC/HBB and the scaling_factor is initially set to 1000.

- **Penalty Term:**
  For slots where the SPOT price is both above the computed threshold and above the 90th percentile (cutoff) of SPOT prices, a penalty term λ×(xi−target)2\lambda \times (x_i - \text{target})^2λ×(xi−target)2 is added to the objective. Here, the target is min⁡(HBB,max_hourly)\min(\text{HBB}, \text{max_hourly})min(HBB,max_hourly) and λ\lambdaλ (lambda_penalty) is set to 10310^3103.

This soft approach allows the optimizer to balance cost minimization and staying close to the desired HBB load without forcing infeasibility.

E. Smoothness Constraint

- **Rationale:**
  To mimic operational constraints (e.g., limited frequency of load changes), we group time slots into 5-hour blocks and add a constraint that adjacent blocks can differ by no more than 5% of TPL.

---

3. Final Code Structure

The final code integrates all the above steps. It is designed to be modular:

- **Data aggregation** is performed in aggregate_weekly_data().
- **Optimization problem** is formulated and solved in load_allocation_optimization().
- **Configuration:**
  You can easily switch between using soft constraints and hard forced constraints by toggling the use_soft_constraint parameter and by selecting the appropriate winsorized data file (df_fixed_winsorized.csv or df_quantile_winsorized.csv).

The code includes detailed diagnostic print statements for each time slot and summaries (such as the SPOT price cutoff, total forced slots, and sum of forced allocations) that help you understand how many slots trigger the forced condition and how they affect the overall solution.

---

4. Results Interpretation

- **Without Forced Constraints:**
  When the forced constraint block is disabled, the solution is feasible but the allocated load in each slot tends to be at the minimum load to reduce cost. The total allocated load (2776.83 MWh) is slightly below the target TPL (2922.98 MWh) due to the relaxed sum constraint.
- **With Soft Constraints:**
  The soft constraint approach introduces a penalty term in the objective when SPOT prices are high. Diagnostic output (per-slot information) indicates that in our synthetic data, most slots do not trigger the penalty because the SPOT prices are

below the high threshold and the 90th percentile cutoff. This approach is more flexible and likely more appropriate when using real-world data.

- **Customization:**
  You will need to tune parameters such as the scaling_factor, lambda_penalty, and the cutoff percentile when real data are available. The current parameters are based solely on synthetic data and our preliminary assumptions based on the given data.

---

5. Next Steps for Real Data Integration

- **Parameter Tuning:**
  With real data, verify the actual distribution of SPOT prices and plant loads. Adjust the hourly bounds, scaling_factor, lambda_penalty, and cutoff percentile accordingly.
- **Model Validation:**
  Validate the model against historical performance and incorporate additional constraints (e.g., ramping limits, production minimums) if needed.
- **Documentation:**
  Maintain a record of the parameters and the rationale for each.

6. Summary

- We have implemented a load allocation optimization model using synthetic data.
- We compared two winsorization approaches (quantile-based and fixed threshold) and integrated the chosen method into the pipeline.
- We designed the optimization problem with bounds, a relaxed total load constraint, and a soft penalty for high SPOT prices.
- Diagnostic outputs are embedded throughout the code to facilitate parameter tuning.
- The model is structured to be easily adjusted for real data by tuning key parameters.