CASE ASSIGNMENT 4

Group 9

Samadrita Roy Chowdhury, Anurag Reddy Katta, Dev Patel,
Laxmi Lalitha Rachana Vepa

Introduction

Case Overview:

- Strategic planning for Giant Motor Company (GMC).
- Focus on optimizing production capacity and flexibility.

Objective:

- Balance production levels for three car lines: Lyra, Libra, Hydra.
- Minimize costs while meeting market demand.

Goals:

1. **Plant Sensitivity Analysis**:
   - Identify plants most affected by demand variability.
   - Determine when specific plants are more likely to open or close based on changing demand.
2. **Demand Variability vs. Profitability**:
   - Explore the relationship between demand uncertainty and profit.
   - Assess whether diversifying production stabilizes or maximizes profit under varying conditions.

Methodology

```python
def solve_with_sampling(DATA, num_samples=1000, force_open_plants=True, verbose=False):
    plants = DATA["Plants"].keys()
    cars = DATA["Demand"].keys()
    productions = [(plant, car) for plant in plants for car in cars]

    # Sampling approach: Generate random demands
    sampled_demands = [
        {car: max(0, int(random.gauss(DATA["Demand"][car], DATA["Demand"][car] * 0.2)))
        for _ in range(num_samples)
    ]

    total_profit = 0
    plant_openings = {plant: 0 for plant in plants}
    diversion_stats = {car: {"diverted": 0, "total_demand": 0} for car in cars}

    # Simulate for each demand sample
    for demand_sample in sampled_demands:
        # Update demand in the data
        sampled_data = DATA.copy()
        sampled_data["Demand"] = demand_sample

        # Solve the model
        m, O, P, D = solve_model(sampled_data, force_open_plants, verbose)
```

```python
# Calculate unfulfilled demand and apply diversions
fulfilled_demand = {car: sum(P[(plant, car)].X for plant in plants) for car in cars}
unfulfilled_demand = {car: max(0, demand_sample[car] - fulfilled_demand[car]) for car in cars}
diverted_demand = {car: 0 for car in cars}

# Apply demand diversion
for car, unfulfilled in unfulfilled_demand.items():
    if unfulfilled > 0:
        for target_car, diversion_rate in DATA["Demand Diversion"][car].items():
            diverted_amount = min(unfulfilled * diversion_rate, demand_sample[car] * diversion_rate)
            diverted_demand[target_car] += diverted_amount
            diversion_stats[target_car]["diverted"] += diverted_amount
```

```
Average Profit: $2608360501.12
Plant Opening Probabilities:
  Lyra: 72.45%
  Libra: 40.05%
  Hydra: 100.00%
  New Lyra: 27.55%
  New Libra: 59.95%
Diversion Statistics:
  Lyra: Diverted 0.0 units (0.00% of total demand)
  Libra: Diverted 250864944.00004357 units (2.27% of total demand)
  Hydra: Diverted 55532199.5002693 units (0.70% of total demand)
```

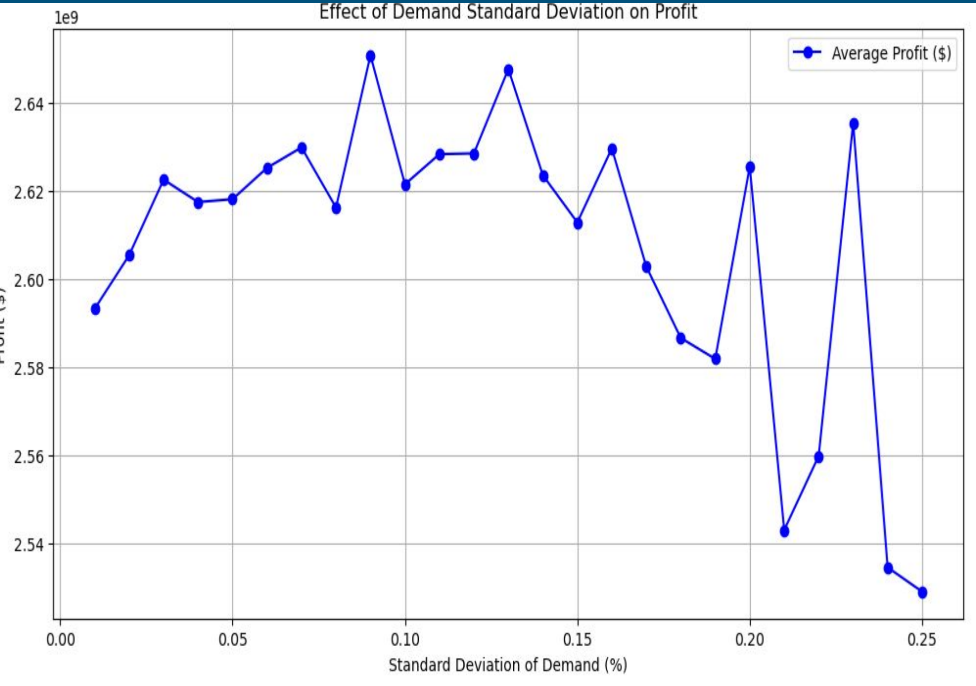
```
Average Profit: $2642632912.55
Plant Opening Probabilities:
  Lyra: 53.43%
  Libra: 50.07%
  Hydra: 100.00%
  New Lyra: 46.57%
  New Libra: 49.93%
Diversion Statistics:
  Lyra: Diverted 0.0 units (0.00% of total demand)
  Libra: Diverted 93578859.00000724 units (0.85% of total demand)
  Hydra: Diverted 20480716.500008076 units (0.26% of total demand)
```

Analysis for different values of standard deviation

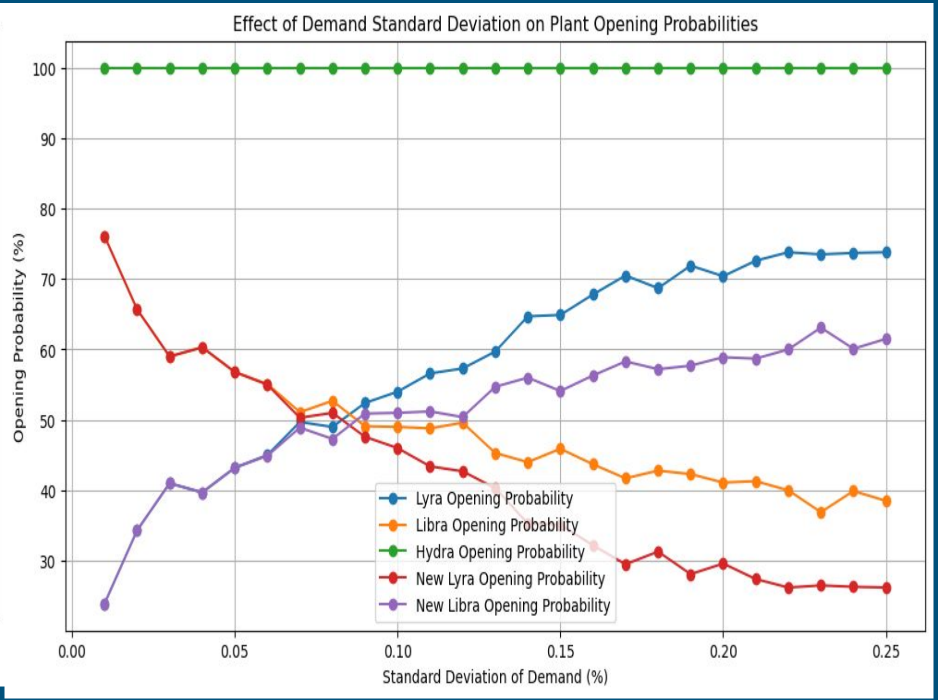
```python
def analyze_std_dev_effect(DATA, std_deviation_range, num_samples=1000, force_open_plants=True):
    std_devs = std_deviation_range  # Percentage standard deviations
    average_profits = []
    plant_opening_probabilities = {plant: [] for plant in DATA["Plants"].keys()}

    for std_dev in std_devs:
        # Generate random demands with current standard deviation
        sampled_demands = [
            {car: max(0, int(random.gauss(DATA["Demand"][car], DATA["Demand"][car] * std_dev))) for car in DATA["Demand"].keys()}
            for _ in range(num_samples)
        ]

        total_profit = 0
        plant_openings = {plant: 0 for plant in DATA["Plants"].keys()}

        for demand_sample in sampled_demands:
            # Update demand in the data
            sampled_data = DATA.copy()
            sampled_data["Demand"] = demand_sample
```

```python
# Define the range of standard deviations to test (1% to 25%)
std_deviation_range = [i / 100 for i in (1,26)]

# Run the analysis
average_profits, plant_opening_probabilities = analyze_std_dev_effect(DATA, std_deviation_range)
```

Results



Effect of Demand Standard Deviation on Profit
1e9
Average Profit ($)
2.64
2.62
2.60
2.58
2.56
2.54
0.00
0.05
0.10
0.15
0.20
0.25
Standard Deviation of Demand (%)



Effect of Demand Standard Deviation on Plant Opening Probabilities
Opening Probability (%)
100
90
80
70
60
50
40
30
Lyra Opening Probability
Libra Opening Probability
Hydra Opening Probability
New Lyra Opening Probability
New Libra Opening Probability
0.00
0.05
0.10
0.15
0.20
0.25
Standard Deviation of Demand (%)

Conclusion

Low to moderate variability (5%-15%) seems manageable and even profitable, suggesting that plants and operations are optimized for such conditions.
Beyond 15%, the system faces challenges that may require revisiting production strategies, capacity planning, or adding flexibility (e.g., increased use of "New Lyra" or "New Libra" plants with higher capacities)