Amazon Air operates a cargo distribution network spanning two primary hubs in the United States. It serves 66 distribution centers worldwide through a combination of direct shipments and strategic routing via focus cities. Transportation costs represent a significant component of operational expenses, making cost optimization essential to maintaining competitive margins. This analysis has come in three parts. We began by analyzing the possible business solutions that could be optimized for this business. We then dove more specifically into Amazon’s operations to establish the mathematical model that made the framework for this optimization.

This concluding analysis optimizes the objective function, which is to minimize total shipping costs, while adhering to the established constraints for operational feasibility. In this analysis, we translate the model previously discussed into a computational solution using Python and the PuLP optimization library. The implementation process involved encoding network data, constructing the linear programming model, solving for optimal flow allocations, and verifying that the solution satisfies all problem requirements. The analysis demonstrates that an optimal solution exists and provides insights into how Amazon Air can minimize distribution costs while meeting customer demand across its global network.

IMPLEMENTATION USING PULP

The optimization problem was implemented in Python using the PuLP linear programming framework. PuLP is an open-source modeling library that provides an intuitive interface for formulating linear and integer programming problems and interfacing with various solvers (Mitchell et al., 2011). The framework allows decision variables, constraints, and objective functions to be expressed in a syntax that closely mirrors mathematical notation, making the implementation both readable and maintainable.

The implementation began by defining the network structure through Python dictionaries containing hubs, focus cities, demand centers, capacity limits, demand requirements, and transportation costs. Decision variables were created for each valid route using PuLP’s LpVariable.dicts() method, with lower bounds set to zero to ensure non-negative flow. There were three variable sets. First were the ‘X’ variables, which defined the flow from hubs to focus cities. Then there were the ‘Y’ variables, defining flow from hubs to centers. Finally, there were the ‘Z’ variables, which defined the flow from the focus cities to the centers. These corresponded directly to the mathematical formulation that had been developed previously.

The objective function minimized total transportation cost by summing the product of unit costs and flow quantities across all routes. Constraints were implemented to enforce

* Demand satisfaction at each center
* Capacity limits at hubs and focus cities
* Flow conservation at focus cities, ensuring that inbound and outbound flows remained balanced.

A GitHub repository containing the complete implementation code and development history is included with this submission.

SOLUTION ANALYSIS AND VERIFICATION

**Constraint Satisfaction**

The solver successfully identified an optimal solution satisfying all problem constraints. Verification was performed programmatically by checking each constraint category against the solution output.

*Hub Capacity Constraints* – Both distribution hubs operated within their monthly capacity limits. Cincinnati/Northern Kentucky (CVG) processed all 95,650 tons against its capacity limit of 95,650 tons, achieving 100% utilization. Alliance Fort Worth (AFW) processed 38,097 tons, with a maximum capacity of 44,350, meaning it was 85.9% utilized. Neither hub exceeded its operational capacity, satisfying the constraints:

For each hub i.

*Demand Satisfaction Constraints* – All 66 distribution centers received exactly their required tonnage. Verification confirmed that for each center *k,* the constraint held.

We ran the following code against the solution to detect any errors in the supplied vs demanded:   
A screen shot of a computer code

AI-generated content may be incorrect.

This confirmed no errors. We ran some samples and found New York received 11,200 and demanded 11,200. Delhi demanded 19,000 and received 19,000. We confirmed that even for the city of Hyderabad, which was converted from a focus center to just a regular distribution center with no demand, its constraint showed no tonnage received and no tonnage demanded.

*Flow Conservation Constraints* – The two active focus cities maintained perfect flow balance. Leipzig received 43,470 tons IN from hubs and distributed exactly 43,470 tons to centers, satisfying the constraint:

San Bernardino, although included in the model, received and distributed zero tons, as the solver determined that direct hub-to-center shipments were more cost-effective than running to San Bernardino first. This was true across all destinations within its service area.

MODEL COMPONENTS:

The complete model implementation included all three required components:

**Decision Variables:**  The solution employed 177 continuous decision variables representing cargo flows: 3 hub-to-focus city routes (), 106 hub-to-center routes (), and 68 focus-to-center routes (). Each variable was constrained to non-negative values, and the solver determined the optimal flow quantities for each route. In our code, we established the decision variables with this:

A screen shot of a computer program

AI-generated content may be incorrect.

**Constraints:** 72 constraints governed the solution space: 66 demand satisfaction constraints ensured that each center received the required tonnage, two hub capacity constraints prevented operational overload, two focus city capacity constraints prevented operational overload at these locations, and two flow conservation constraints maintained balance at focus cities. The constraints were defined with the following pieces of code:

A screenshot of a computer screen

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

**Objective Function:** The objective function is minimized through the expression:

The cost coefficients range from $0.25 to $1.60 per ton, depending on route distance and operational characteristics. In our code, we included the objective function with this code:

A screenshot of a computer code

AI-generated content may be incorrect.

EXPLAINING EXPECTED FROM ACTUAL OUTCOMES

The solver’s output aligns with expected optimization behavior given the cost structure and network topology. Several key patterns emerge:

*Leipzig Focus City Dominance:* All Indian demand (43,470 tons total) routes through Leipzig, representing the only feasible path to reach Bengaluru, Coimbatore, Delhi, and Mumbai. The model correctly identified that despite the multi-hop cost (Cincinnati to Leipzig at $1.50, then Leipzig to Indian centers at $1.50), this represents the only viable routing option for this demand segment.

*San Bernardino Non-Utilization:* The solver determined that routing cargo through San Bernardino offered no cost advantage. Direct hub-to-center shipments typically cost $0.50 per ton, while routing through San Bernardino would incur $0.50 to reach the focus city, plus an additional $0.50 to $0.70 to reach final destinations, making the total cost higher or equal. The optimization correctly bypassed this intermediate node in favor of direct shipments.

*Hub Load Distribution:* CVG handles the majority of outbound cargo due to its superior access to European and Indian markets via Leipzig. AFW serves primarily domestic U.S. markets, where its geographic positioning offers cost advantages, particularly for Texas destinations, where unit costs drop to $0.25 per ton. This load distribution reflects the underlying cost matrix and demonstrates the solver’s ability to exploit geographic efficiencies.

The solution’s total cost of $199,476.25 represents the minimum achievable cost given the network structure, capacity constraints, and demand requirements, confirming successful optimization.

REFLECTION ON DEVELOPMENT PROCESS

The development of this optimization model is closely aligned with established linear programming methodologies, while revealing several practical challenges characteristic of real-world operations research applications. The process followed the structured approach outlined by Williams (2013), progressing from problem formulation through mathematical modeling, computational implementation, and solution validation.

**Data Preparation Challenges**

The initial phase presented unexpected obstacles related to data quality and consistency. The source documentation contained naming inconsistencies (e.g., “Rockford” versus “Chicago/Rockford”, “Bangalore” versus “Bangaluru”) that required manual reconciliation across multiple data structures. This experience reinforced Winston’s (2022) observation that data preparation often consumes more time than model construction in practical optimization projects. Additionally, the ambiguous classification of Hyderabad, initially listed as both a focus city and lacking direct hub connectivity, necessitated an improvised solution to avoid this inconsistency.

**Model Formulation Insights:**

The translation from mathematical notation to executable code using PuLP proved more intuitive than anticipated. The framework’s dictionary-based approach for decision variables enabled a natural mapping between route tuples and flow quantities, maintaining semantic clarity throughout the implementation (Mitchell et al., 2011). However, the initial model formulation contained a critical error: the focus city capacity constraints incorrectly double-counted throughput by summing both inbound and outbound flows. This resulted in an infeasible solution, demonstrating how subtle specification errors can render otherwise valid models unsolvable. The debugging process highlighted the importance of verifying constraints before attempting solutions, a principle emphasized in the optimization literature (Hillier & Lieberman, 2021).

**Unexpected Solution Characteristics**

The optimal solution revealed operational insights not immediately apparent from the problem structure. San Bernardino's focus city remained completely unutilized, with all cargo routing either directly from hubs or through Leipzig. This outcome, while mathematically optimal given the cost structure, suggests potential inefficiencies in the current network design. In a real-world scenario, this finding would prompt questions about San Bernardino’s strategic value and whether capacity investments there represent a misallocation of resources. The concentration of all Indian demand through Leipzig (43,470 tons) also identifies a critical bottleneck and single point of failure in the distribution network.

**Process Expectations Versus Reality**

The most significant deviation from initial expectations involved the number of iteration cycles. The assumption that a correctly formulated mathematical model would translate smoothly into working code proved overly optimistic. Multipole iterations addressed data inconsistencies, constraint formulation errors, and validation requirements. This iterative refinement process mirrors industry practice, where model development rarely proceeds linearly from specification to solution (Rardin, 2016). The experience underscored that optimization modeling is inherently iterative, requiring continuous validation at each development stage.

The development process ultimately confirmed that successful optimization modeling requires equal attention to problem formulation, data quality, implementation details, and solution interpretation. While the mathematical foundations provide necessary structure, practical success depends on careful validation, iterative refinement, and critical evaluation of results within the broader business context.

Hillier, F. S., & Lieberman, G. J. (2021). *Introduction to Operations Research* (11th ed). McGraw-Hill Education.

Mitchell, S., O’Sullivan, M., & Dunning, I. (2011). PuLP: A linear programming toolkit for Python. *The University of Auckland.*

Rardin, R. L. (2016). *Optimization in operations research* (2nd ed.). Pearson.

Williams, H.P. (2013). *Model building in mathematical programming* (5th ed.). John Wiley & Sons.

Winston, W.L. (2022). *Operations research: Applications and algorithms* (5th ed.). Cengage Learning.