International Journal of Applied Engineering Research ISSN 0973-4562 Volume 19, Number 4 (2024) pp. 340-347

Efficient MATLAB Algorithms for Solving Transportation Problem: From Initial to Optimal Solutions

[1]Hilal A. Abdelwali, [1]Farraj Aldaihani, [1]Swilem A.M. Swilem, [2]Mohamed H. Abdelati
[1]Assistant Professor, Automotive and Marine Department, College of Technological Studies, PAAET, Kuwait.
[2]Automotive and Tractors Dept. Faculty of Engineering, Minia University, Egypt.

Abstract
The transportation problem is solved usually in three successive steps. These steps include: Finding the initial basic feasible solution, test of optimality, and moving towards optimality. There are many different methods that can be used to find the initial basic feasible solution. Out of these methods are: North-West Corner Method, Row Minima Method, Column Minima Method, Least Cost Method, Vogel’s Approximation Method, and many other methods. There are two other famous methods that can be used for the test of optimality and moving towards optimality steps. These two methods are the Stepping Stone Method, and the Modified Distribution (MODI) method. The transportation problem can be solved by preparing computer programs using any programming language compilers based on the three solution steps. There are ready-to-go packages to solve the transportation problem like QM manager, TORA, and many others. But most of these packages are expensive. So, they are not affordable to be used by small transportation companies. This paper presents a comprehensive framework for solving the transportation problems using MATLAB, explicitly focusing on obtaining the initial basic feasible solutions and optimizing them using the Modified Distribution Method (MODI). MATLAM is less expensive and affordable to be got and use. The research highlights some of the well-established methods, each of them is implemented in MATLAB to derive initial solutions and to enhance these solutions, to minimize the total transportation costs. The MATLAB scripts were validated against Lingo software results as well as manually solved examples, demonstrating their accuracy and efficiency in solving complex transportation problems. The results underline the practical applications of these MATLAB scripts in automotive engineering logistics, providing a more accessible and precise alternative to specialized software, ultimately facilitating cost-effective and efficient transportation networks.

1. Introduction

The transportation problem remains relevant and important in automotive engineering, logistics, and transportation because it influences the overall transportation cost and the flow of products[1, 2]. In the modern world more and more value are placed on effectiveness and feasibility of the solutions offered in the sphere of transportation, which are crucial for effective supply chain and planning. The optimization of the cost, time and distance is crucial in the improvement of the performance of business entities[3-5].

Researchers have delved into various methods to tackle the transportation problem. Among these methods are the North-west Corner Method (NWCM), which commences from the north-west corner of the distribution table and assigns quantities until supply and demand are balanced, the Row Minima Method (RMM), which concentrates on selecting the least cost in each row, and the Column Minima Method (CMM), which aims for the least cost in each column. The Least Cost Method (LCM) opts for the overall least cost in the distribution table. At the same time, Vogel's Approximation Method (VAM) is renowned for its efficiency, calculating the differences between the lowest two costs in each row and column to identify the best distribution[6].

Some critical papers in this area include the work by Hussein et al [7] which introduced a modification to VAM that offered enhanced primal solutions at the commencement of the transportation problem. Some of the studies they conducted revealed that this social approach was indeed a better way to go. Likewise[2], Hosseini [8] composed an operational research with MATLAB concerning enormous transportation issues while asserting the better performance of newly proposed algorithms. Another important work of Ishaq et al [9] provided a computational model to solve the transportation problem that was incorporated in software form. This model was more efficient in its solutions than those procured by hand calculations. In addition, Wu et al. [10] expressed their interest in applying the MATLAB technique at optimizing the traffic light timings, so as to enhance the traffic flow of the cities and decongest them[11].

The latest developments are not entirely efficient since most of the specialized software packages like TORA and QM. LINGO, LINDO and more other compilers are expensive to get and need some expertise in that field before using them comfortably in most of their practices[12]. Moreover, in this field, there is not much work done to utilize well-known, less expensive and mighty software such as MATLAB for algorithm computation and solving sophisticated issues[13, 14].

This study aims at designing and implementing MATLAB codes (scripts) for obtaining the initial basic feasible solution and the optional solution of the transportation using the modified distribution method (MODI) for transportation problem based on the solution strategies in [15, 16]. Using such scripts, a wider range of professionals involved in automotive engineering, logistics and transport may solve the various transpiration problems with less effort and in a shorter amount of time[17, 18]. These scripts will offer more accurate replacement of software for the transportation sector, filling the gap and helping companies in minimizing costs and rendering their transportation networks more efficient, sustainable and profitable[19, 20].

2. Methodology

This section outlines the methodologies used to solve the transportation model using MATLAB. The actual MATLAB scripts of the different methods are included in this section as well. We shall kickstart the first solutions, then explain in detail the MODI Method, which we used to attain the optimal solution, together with a brief description of each of our methods for the algorithm [21].

Other slight changes were made to the solution algorithms well-known for the transportation problem to fit within the developed codes in MATLAB. As such, allowances were created so that each algorithm obtained the optimal structure in MATLAB, which increases computational efficiency and accuracy. Modifying the algorithms facilitates embedding in MATLAB and allows their extension of relevance in practice toward the solution of more complex transportation problems in a more simplistic manner[22].

Initial Basic Feasible Solution Techniques [23-27]

2.1 North-West Corner Method

The North-West Corner Method is an algorithm for obtaining an initial basic feasible solution for the transportation problem. The procedure starts in the top-left (north-west) cell of the cost matrix and fills as much allocations as possible to the shipping routes, then moves on to the next cell in a very systemic way until all demands and supplies are satisfied.

The proposed solution algorithm for this method is:

1. Start
2. Initialize: Set the initial position to the northwest corner (cell (1,1)).
3. Allocate Quantity: Allocate X11 = min(a1, b1) to cell (1,1).
   - If b1 > a1: Move vertically down to the second row.
- o Allocate X21 = min(a2, b1 - X11).
- If b1 < a1: Move horizontally right to the second column.
- o Allocate X12 = min(a1 - X11, b2).
   - If b1 = a1: There is a tie. Choose either cell (1,2) or (2,1) for the next allocation.
4. Repeat: Move to the new northwest corner of the remaining sub-matrix.
   - Repeat the allocation process until all requirements are satisfied.
5. End

2.2 Row Minima Method

The Row Minima Method focuses on securing the minimum cost in each row and gives as much allocations as possible to that route. Similarly, the process is carried on for the remaining rows till all the demands and supplies are fulfilled.

The proposed solution algorithm for this method is:

1. Start
2. Initialize: Set the initial position to the first row.
3. Choose Minimum Cost Cell: In the current row, choose the cell with the minimum cost.
4. Allocate Quantity: Allocate Xij = min(ai, bj) to the chosen cell.
   - If ai <= bj: Move to the next row.
   - If ai > bj: Move to the next column in the same row.
5. Repeat: Move to the new minimum cost cell in the next row or column.
   - o Repeat the allocation process until all requirements are satisfied.
6. End

2.3 Column Minima Method

The Column Minima Method focuses on the lowest possible cost in one column. Commodity previously allocated on that route is then assigned as much as possible. This process is repeated for all columns until all the demands and supplies are satisfied.

The proposed solution algorithm for this method is:

1. Start
2. Initialize: Set the initial position to the first column.
3. Choose Minimum Cost Cell: In the current column, choose the cell with the minimum cost.
4. Allocate Quantity: Allocate Xij = min(ai, bj) to the chosen cell.
   a. If ai <= bj: Move to the next row in the same column.
   b. If ai > bj: Move to the next column.
5. Repeat: Move to the new minimum cost cell in the next column or row.
6. Repeat the allocation process until all requirements are satisfied.
7. End

2.4 Least Cost Method

The Least-Cost Method tries to minimize the total cost of transportation by choosing the lowest price available and filling as much allocations as possible on that particular route. In the same way, the process is carried on until all the demands and supplies are fulfilled.

The proposed solution algorithm for this method is:

1. Start
2. Initialize: Identify the cell with the lowest unit cost in the matrix.
3. Allocate Quantity: Allocate Xij = min(ai, bj) to the chosen cell.
4. Check Conditions:
   - If ai <= bj: Mark the row as exhausted and move to the following minimum cost cell.
   - If ai > bj: Mark the column as exhausted and move to the following minimum cost cell.
5. Repeat: Continue identifying the cell with the lowest cost and allocate quantities until all supplies and demands are satisfied.
6. End

2.5 Vogel's Approximation Method

Vogel's Approximation Method takes all costs in each row and each column and then computes the penalty for not using the lowest price of the route. It then sorts the route with the highest penalty whenever practicable is assigned. The above process runs again until all demands and supplies are satisfied and exhausted.

The proposed solution algorithm for this method is:

1. Start
2. Calculate Penalties:
   - For each row, find the smallest and the second smallest costs. Calculate the difference (penalty).

- For each column, find the smallest and the second smallest costs. Calculate the difference (penalty).

3. Identify the Largest Penalty: Identify the row or column with the largest penalty.
4. Allocate Quantity: Choose the cell with the lowest cost in the identified row or column. Allocate Xij = min(ai, bj) to the chosen cell.
5. Update Supply/Demand:
   - If ai <= bj: Mark the row as exhausted and remove it from further consideration.
   - If ai > bj: Mark the column as exhausted and remove it from further consideration.
6. Recalculate Penalties: Recalculate the penalties for the remaining rows and columns.
7. Repeat steps 3 to 6 until all supplies and demands are satisfied.
8. End

2.6 Optimal Solution: MODI Method (Modified Distribution Method)

The MODI Method obtains the optimal solution by enhancing the first feasible algorithm process. It finds the opportunity cost of every unutilized assignment and then attempts to re-assign it to minimize the total price.

The proposed solution algorithm for this method is:

1. Start
2. Find an Initial Basic Feasible Solution: Solve the transportation problem using any method to get an initial basic feasible solution.
3. Calculate ui and vj: For each occupied cell, calculate the row (ui) and column (vj) potentials using the equation cij = ui + vj. Assume u1 = 0 to start.
4. Calculate Opportunity Cost: For each unoccupied cell, calculate the opportunity cost Δij = cij - ui - vj.
5. Check for Optimality:
   - If all Δij >= 0, the current solution is optimal. Go to step 8.
   - If any Δij < 0, proceed to the next step to improve the solution.
6. Identify Entering Variable: Identify the cell with the most negative Δij. This will be the entering variable.
7. Form Closed Loop and Adjust Allocations: Form a closed loop with the entering variable and the occupied cells. Assign alternate plus and minus signs starting with the entering variable. Determine the maximum feasible amount to be transferred (theta) and adjust the allocations along the loop.
8. Update Solution: Update the solution by adding theta to the entering variable and adjusting the other variables in the loop accordingly.
9. Repeat: Repeat steps 3 to 8 until the optimality condition is satisfied.
10. End

3. Implementation

Figures (1 - 6) below provide the MATLAB scripts for every method covered in the Methodology section. Replicating this study's findings requires these scripts. They include the Modi Method for determining the optimal solution and implementations of several initial solution strategies.

All of the MATLAB scripts are included as images in this document for clarity and accuracy. The following scripts cover:

3.1 North-West Corner Method

Figure (1): The North-West Corner Method's MATLAB script begins in the top-left cell and methodically allocates quantities until supply and demand are balanced.

3.2 Row Minima Method

Figure (2): The main goals of the MATLAB script for the Row Minima Method are securing the lowest cost in each row and allocating as much allocations as possible to that route.

3.3 Column Minima Method

Figure (3): The Column Minima Method's MATLAB script repeats the process for each column until all demands and supplies are met, concentrating on the lowest cost in one column at a time.

3.4 Least Cost Method

Figure (4): The MATLAB script for the least-cost method seeks to minimize the overall cost of transportation by selecting the lowest price and filling as much as possible on that specific route.

3.5 Vogel's Approximation Method

Figure (5): To reduce overall transportation costs, quantities are distributed according to the MATLAB script for Vogel's Approximation Method, which also computes penalties for not taking the lowest-cost routes.

3.6 MODI Method (Modified Distribution Method)

Figure (6): The MODI Method's MATLAB script improves the first feasible solution procedure, computes opportunity costs, and modifies allocations to minimize overall costs to arrive at the optimal solution.

4. Explanation of Key Parts of the Scripts:

Initialization and Allocation: The first steps in any method are to initialize the position and allocate quantities according to predetermined guidelines or standards (such as minimum cost or penalties).

Loops and Conditions: The scripts use conditions to determine whether supply and demand constraints are met and loops to repeatedly cycle through rows, columns, or cells.

Checking Optimality (MODI Method): The Modified Distribution (MODI) Method includes extra steps such as completing opportunity cost calculations, ensuring optimality, and making adjustments to allocations through a closed loop. The following Figures (1) to (6) illustrate the prepared MATAL scripts for the mentioned methods.

```
% Get user input for supply vector
supply = input('Enter the supply vector: ');

% Get user input for demand vector
demand = input('Enter the demand vector: ');

% Get user input for cost matrix
cost = input('Enter the cost matrix: ');

% Initialize the transportation matrix with zeros
solution = zeros(length(supply), length(demand));

% Initialize indices for supply and demand vectors
supplyIndex = 1;
demandIndex = 1;

% Apply North-West Corner Method
while supplyIndex <= length(supply) && demandIndex <= length(demand)
    % Determine the quantity to be transported from current supply and demand
    quantity = min(supply(supplyIndex), demand(demandIndex));

    % Assign the quantity to the corresponding cell in the solution matrix
    solution(supplyIndex, demandIndex) = quantity;

    % Update supply and demand vectors based on the assigned quantity
    supply(supplyIndex) = supply(supplyIndex) - quantity;
    demand(demandIndex) = demand(demandIndex) - quantity;

    % Move to the next supply or demand location if one of them is exhausted
    if supply(supplyIndex) == 0
        supplyIndex = supplyIndex + 1;
    else
        demandIndex = demandIndex + 1;
    end
end

% Calculate the total transportation cost (objective value)
totalCost = sum(sum(solution .* cost));

% Display the solution matrix obtained using NWCM
disp('Solution Matrix using NWCM:');
disp(solution);

% Display the total transportation cost (objective value)
disp('Total Transportation Cost:');
disp(totalCost);
```

Figure 1: MATLAB script for NWCM.

```
% Get user input for supply vector
supply = input('Enter the supply vector: ');

% Get user input for demand vector
demand = input('Enter the demand vector: ');

% Get user input for cost matrix
cost = input('Enter the cost matrix: ');

% Initialize the transportation matrix with zeros
solution = zeros(length(supply), length(demand));

% Store the original cost matrix for final cost calculation
originalCost = cost;

% Apply Row Minima Method
for i = 1:length(supply)
    while supply(i) > 0 && any(demand > 0)
        % Find the minimum cost in the current row
        [minCost, col] = min(cost(i, :));

        % Determine the quantity to be transported from current supply and demand
        quantity = min(supply(i), demand(col));

        % Assign the quantity to the corresponding cell in the solution matrix
        solution(i, col) = quantity;

        % Update supply and demand vectors based on the assigned quantity
        supply(i) = supply(i) - quantity;
        demand(col) = demand(col) - quantity;

        % If a demand is exhausted, set the corresponding costs to infinity
        if demand(col) == 0
            cost(:, col) = inf;
        end
    end
end

% Calculate the total transportation cost (objective value) using the original cost matrix
totalCost = sum(sum(solution .* originalCost));

% Display the solution matrix obtained using Row Minima Method
disp('Solution Matrix using Row Minima Method:');
disp(solution);

% Display the total transportation cost (objective value)
disp('Total Transportation Cost:');
disp(totalCost);
```

Figure 2: MATLAB script for Row Minima Method.

```
% Get user input for supply vector
supply = input('Enter the supply vector: ');

% Get user input for demand vector
demand = input('Enter the demand vector: ');

% Get user input for cost matrix
cost = input('Enter the cost matrix: ');

% Initialize the transportation matrix with zeros
solution = zeros(length(supply), length(demand));

% Store the original cost matrix for final cost calculation
originalCost = cost;

% Apply Column Minima Method
for j = 1:length(demand)
    while demand(j) > 0 && any(supply > 0)
        % Find the minimum cost in the current column
        [minCost, row] = min(cost(:, j));

        % Determine the quantity to be transported from current supply and demand
        quantity = min(supply(row), demand(j));

        % Assign the quantity to the corresponding cell in the solution matrix
        solution(row, j) = quantity;

        % Update supply and demand vectors based on the assigned quantity
        supply(row) = supply(row) - quantity;
        demand(j) = demand(j) - quantity;

        % If a supply is exhausted, set the corresponding costs to infinity
        if supply(row) == 0
            cost(row, :) = inf;
        end
    end
end

% Calculate the total transportation cost (objective value) using the original cost matrix
totalCost = sum(sum(solution .* originalCost));

% Display the solution matrix obtained using Column Minima Method
disp('Solution Matrix using Column Minima Method:');
disp(solution);

% Display the total transportation cost (objective value)
disp('Total Transportation Cost:');
disp(totalCost);
```

Figure 3: MATLAB script for Column Minima Method.

```
% Get user input for supply vector
supply = input('Enter the supply vector: ');

% Get user input for demand vector
demand = input('Enter the demand vector: ');

% Get user input for cost matrix
cost = input('Enter the cost matrix: ');

% Initialize the transportation matrix with zeros
solution = zeros(length(supply), length(demand));

% Store the original cost matrix for final cost calculation
originalCost = cost;

% Apply Least Cost Method
while any(supply > 0) && any(demand > 0)
    % Find the minimum cost in the entire cost matrix
    [minCost, index] = min(cost(:));
    [row, col] = ind2sub(size(cost), index);

    % Determine the quantity to be transported from current supply and demand
    quantity = min(supply(row), demand(col));

    % Assign the quantity to the corresponding cell in the solution matrix
    solution(row, col) = quantity;

    % Update supply and demand vectors based on the assigned quantity
    supply(row) = supply(row) - quantity;
    demand(col) = demand(col) - quantity;

    % If a supply is exhausted, set the corresponding row costs to infinity
    if supply(row) == 0
        cost(row, :) = inf;
    end

    % If a demand is exhausted, set the corresponding column costs to infinity
    if demand(col) == 0
        cost(:, col) = inf;
    end
end

% Calculate the total transportation cost (objective value) using the original cost matrix
totalCost = sum(sum(solution .* originalCost));

% Display the solution matrix obtained using Least Cost Method
disp('Solution Matrix using Least Cost Method:');
disp(solution);

% Display the total transportation cost (objective value)
disp('Total Transportation Cost:');
disp(totalCost);
```

Figure 4: MATLAB script for Least Cost Method.

```
% Get user input for supply vector
supply = input('Enter the supply vector: ');

% Get user input for demand vector
demand = input('Enter the demand vector: ');

% Get user input for cost matrix
cost = input('Enter the cost matrix: ');

% Initialize the transportation matrix with zeros
solution = zeros(length(supply), length(demand));

% Store the original cost matrix for final cost calculation
originalCost = cost;

% Apply Vogel Approximation Method
while any(supply > 0) && any(demand > 0)
    % Calculate penalties for rows and columns
    rowPenalties = zeros(length(supply), 1);
    colPenalties = zeros(length(demand), 1);

    for i = 1:length(supply)
        if supply(i) > 0
            sortedCosts = sort(cost(i, :));
            rowPenalties(i) = sortedCosts(2) - sortedCosts(1);
        else
            rowPenalties(i) = -inf; % Ignore exhausted rows
        end
    end

    for j = 1:length(demand)
        if demand(j) > 0
            sortedCosts = sort(cost(:, j));
            colPenalties(j) = sortedCosts(2) - sortedCosts(1);
        else
            colPenalties(j) = -inf; % Ignore exhausted columns
        end
    end

    % Find the maximum penalty
    [maxRowPenalty, rowIdx] = max(rowPenalties);
    [maxColPenalty, colIdx] = max(colPenalties);

    % Determine whether to use row or column penalty
    if maxRowPenalty >= maxColPenalty
        % Allocate based on row penalty
        [minCost, col] = min(cost(rowIdx, :));
        row = rowIdx;
    else
        % Allocate based on column penalty
        [minCost, row] = min(cost(:, colIdx));
        col = colIdx;
    end

    % Determine the quantity to be transported from current supply and demand
    quantity = min(supply(row), demand(col));

    % Assign the quantity to the corresponding cell in the solution matrix
    solution(row, col) = quantity;

    % Update supply and demand vectors based on the assigned quantity
    supply(row) = supply(row) - quantity;
    demand(col) = demand(col) - quantity;

    % If a supply or demand is exhausted, set the corresponding costs to infinity
    if supply(row) == 0
        cost(row, :) = inf;
    end
    if demand(col) == 0
        cost(:, col) = inf;
    end
end

% Calculate the total transportation cost (objective value) using the original cost matrix
totalCost = sum(sum(solution .* originalCost));

% Display the solution matrix obtained using VAM
disp('Solution Matrix using Vogel Approximation Method:');
disp(solution);

% Display the total transportation cost (objective value)
disp('Total Transportation Cost:');
disp(totalCost);
```

Figure 5: MATLAB script for Vogel's Approximation Method.

```
% Get user input for supply vector
supply = input('Enter the supply vector: ');

% Get user input for demand vector
demand = input('Enter the demand vector: ');

% Get user input for cost matrix
cost = input('Enter the cost matrix: ');

% Flatten the cost matrix into a column vector
c = cost(:);

% Define the constraint matrices
Aeq = zeros(numel(demand) + numel(supply), numel(c));
beq = zeros(numel(demand) + numel(supply), 1);

% Constraint: Total supply from factories must meet total demand at warehouses
for i = 1:numel(demand)
    Aeq(i, (i-1)*numel(supply) + 1 : i*numel(supply)) = 1;
    beq(i) = demand(i);
end

% Constraint: Total demand at warehouses must meet total supply from factories
for i = 1:numel(supply)
    Aeq(numel(demand) + i, i : numel(supply) : numel(c)) = 1;
    beq(numel(demand) + i) = supply(i);
end

% Bounds: Variables must be non-negative
lb = zeros(size(c));
ub = [];

% Solve the linear programming problem
[x, fval] = linprog(c, [], [], Aeq, beq, lb, ub);

% Reshape the solution vector into the transportation matrix
solution = reshape(x, size(cost));

% Display the optimal solution
disp('Optimal Solution:')
disp(solution)
disp('Total Cost:')
disp(fval)
```

Figure 6: MATLAB script for Modified Distribution Method.

5. Code Validation

LINGO and manually solved examples were used to validate the MATLAB scripts to guarantee their accuracy and dependability. To ensure that the MATLAB scripts produce the same results as the reliable Lingo software and the manually solved examples, the same example problems were run in MATLAB. Some of problems were solved manually for more check. The results were then compared. This validation procedure shows how well the MATLAB implementations work and how well they can be applied to real-world transportation issues[28, 29].

6. Results and Discussion

We solved many example problems and case studies, with differences in the initial feasible basic solutions generated from each method using the prepared MATLAB scripts. Figures (1) to (6) walk through the solution algorithms and codes step by step, showing the allocations and the corresponding costs. While this shows the difference in initial constraints, you might start with depending on the chosen approach.

After Applying the MODI Method, We got the Best Solution. It then improves (refines) the initial basic feasible solutions by computing the opportunity cost of each unused assignment and reassigning them to minimize the total transportation cost. The Figures shown illustrate the desired optimal allocations and costs for the final solution or the solution obtained after employing the MODI Method, and therefore, the resultant reduced overall cost and improved efficiency. Figure (7) to (11) shows the MATLAB scripts outputs versus manually solved example for NWCM, Row Minima Method, Column Minima Method, Least Cost Method and Vogel's Approximation Method. Figure (12) compares the MATLAB output versus the LINGO code output for the same example. As shown in Figures, we got the exact results for all methods. These Figure are good proof of validation.

Lingo was compared with MATLAB scripts to verify the validity of the MATLAB MODI script. Results obtained from MATLAB coincided with results from Lingo and from the manually solved problems, ensuring the accuracy of solutions concerning the MATLAB implementations used here. These Figures include side-by-side comparisons of the MATLAB scripts, manually solved and Lingo code results to emphasize the agreement between the different techniques.

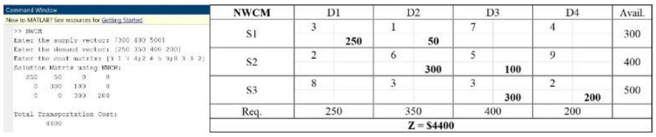

Figure 7: MATLAB output versus manually solved example for NWCM.

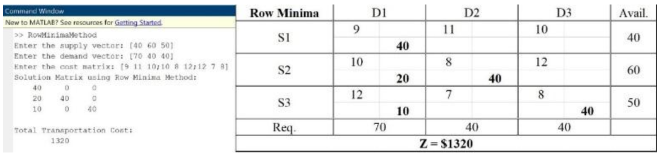

Figure 8: MATLAB output versus manually solved example for Row Minima Method.

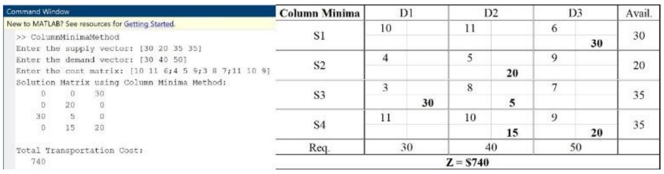

Figure 9: MATLAB output versus manually solved example for Column Minima Method.

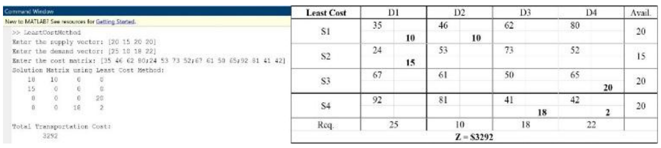

Figure 10: MATLAB output versus manually solved example for Least Cost Method.

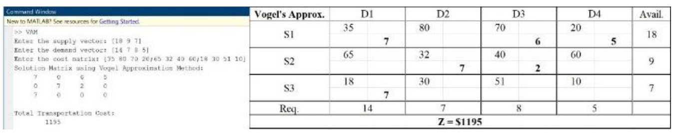

Figure 11: MATLAB output versus manually solved example for Vogel's Approx. Method.

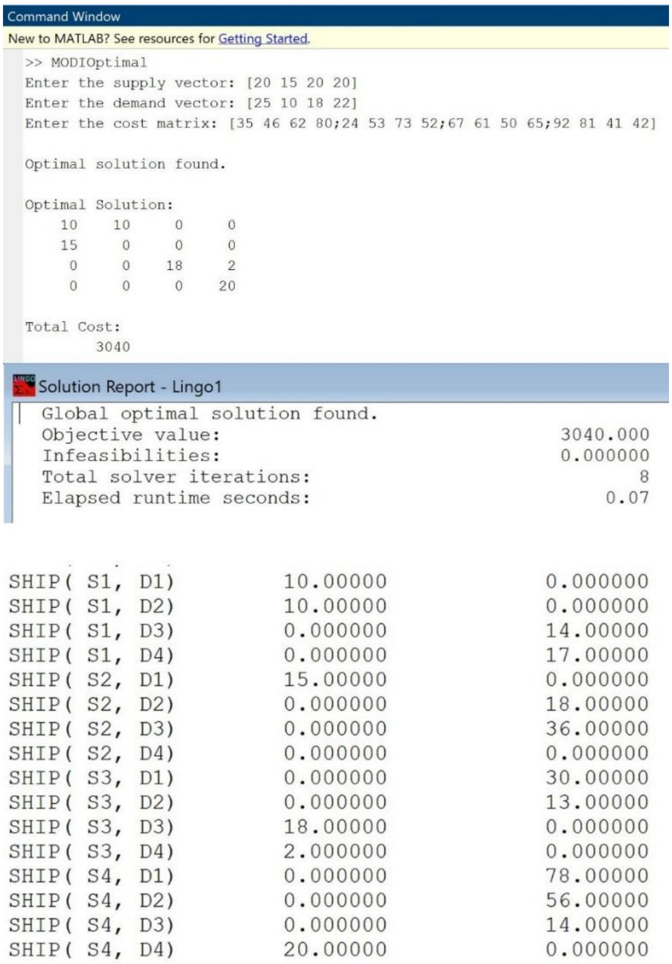

Figure 12: MATLAB output versus LINGO results for the MODI (Optimal Solution).

7. Conclusion and Recommendation

This research successfully implements and validates a set of MATLAB algorithms for addressing transportation problems, providing significant contributions to the field of automotive engineering logistics. By comparing different initial basic feasible solution methods and refining these solutions using the MODI method. The study demonstrates the effectiveness of MATLAB in optimizing transportation costs. The comparison with Lingo software as well as the manually solved problems validates the accuracy and reliability of the MATLAB implementations, ensuring that they can handle real-world transportation issues with precision.

The study's findings emphasize the potential of MATLAB scripts to serve as robust tools for professionals in logistics and transportation, offering a user-friendly alternative to more complex software. The implemented methods simplify the computation process and enhance the overall efficiency and sustainability of transportation networks. Future work may explore further refinements and extensions of these algorithms and their application to other logistical challenges, thereby broadening the scope and impact of this research in the field of supply chain optimization.

References

[1] H. A. Taha, Operations research: an introduction. Pearson Education India, 2013.

[2] M. Hilal Abdelatli Abdelwali, "A new approach for finding an initial basic feasible solution to a transportation problem," Journal of Advanced Engineering Trends, vol. 43, no. 1, pp. 77-85, 2024.

[3] W. L. Winston, Operations research: applications and algorithms. Cengage Learning, 2022.

[4] C. Krishnaraj, A. A. Jayakumar, and S. D. Shri, "Solving supply chain network optimization models using LINGO," International Journal of Applied Engineering Research, vol. 10, no. 19, pp. 14715-14718, 2015.

[5] S. H. Zegordi, I. K. Abadi, and M. B. Nia, "A novel genetic algorithm for solving production and transportation scheduling in a two-stage supply chain," Computers & industrial engineering, vol. 58, no. 3, pp. 373-381, 2010.

[6] A. El-Wahab, M. Rabie, M. H. Abdelati, M. I. Khalil, and K. Abdelgawwad, "A case study of reducing the total wasted time for the bus movement of Public Transportation Authority in Cairo (CTA)," SVU-International Journal of Engineering Sciences and Applications, vol. 2, no. 2, pp. 82-87, 2021.

[7] H. Hussein, M. A. Shiker, and M. S. Zabiba, "A new revised efficient of VAM to find the initial solution for the transportation problem," in Journal of Physics: Conference Series, 2020, vol. 1591, no. 1: IOP Publishing, p. 012032.

[8] E. Hosseini, "Three new methods to find initial basic feasible solution of transportation problems," Applied Mathematical Sciences, vol. 11, no. 37, pp. 1803-1814, 2017.

[9] A. M. Maatuk, H. S. Idrees, and E. H. Ali, "An optimal solution for transportation problem using computing modelling," in 2016 International Conference on Engineering & MIS (ICEMIS), 2016: IEEE, pp. 1-5.

[10] W. Jianhua, L. Ning, X. Enhong, and Z. Xin, "Optimal design for traffic light and simulation based on MATLAB," in 2010 International Conference on Mechanic Automation and Control Engineering, 2010: IEEE, pp. 2854-2857.

[11] G. Sharma, S. Abbas, and V. Gupta, "Solving transportation problem with the various method of linear programming problem," Asian journal of current engineering and maths, vol. 1, no. 3, pp. 81-83, 2012.

[12] M. Alardhi, H. A. Abdelwali, A. M. Khalfan, and M. H. Abdelati, "Using the Minimize Distance Method to Find the Best Compromise Solution of Multi-objective Transportation Problem with Case Study."

[13] E. Goodarzi, M. Ziaei, E. Z. Hosseinipour, E. Goodarzi, M. Ziaei, and E. Z. Hosseinipour, "Optimization analysis using LINGO and MATLAB," Introduction to Optimization Analysis in Hydrosystem Engineering, pp. 149-193, 2014.

[14] M. H. Abdelati, A. M. Abd-El-Tawwab, E. E. M. Ellimony, and M. Rabie, "" Efficient and Versatile Methodology for Solving Solid Transportation Problem Using Excel Solver: A Comparative Study with LINGO Code," Industrial Engineering, vol. 3, no. 1, 2024.

[15] G. Guo and J. Obłój, "Computational methods for martingale optimal transport problems," The Annals of Applied Probability, vol. 29, no. 6, pp. 3311-3347, 2019.

[16] R. G. Patel, B. S. Patel, and P. Bhathawala, "On optimal solution of a transportation problem," Global Journal of Pure and Applied Mathematics, vol. 13, no. 9, pp. 6201-6208, 2017.

[17] S. Aramuthakannan and P. Kandasamy, "Revised distribution method of finding optimal solution for transportation problems," IOSR Journal of Mathematics, vol. 4, no. 5, pp. 39-42, 2013.

[18] M. H Abdelati and A. Abdelhafeez, "An Integrated Approach for Promoting Sustainable Transportation: A Case Study of Fuel Consumption and Carbon Dioxide Emissions Reduction in the Cairo Transport Authority," International Journal of Advanced Engineering and Business Sciences, vol. 4, no. 3, 2023.

[19] Y. A. Hussein and M. A. Shiker, "Using a new method named matrix method to find the optimal solution to transportation problems," in AIP Conference Proceedings, 2023, vol. 2414, no. 1: AIP Publishing.

[20] V. Vidhya and K. Ganesan, "An Alternate Method for Finding Optimal Solution to Solid Transportation Problem under Fuzzy Environment," IOP Conference Series: Materials Science and Engineering, vol. 912, no. 6, p. 062047, 2020, doi: 10.1088/1757-899x/912/6/062047.

[21] G. Dantzig, "Linear programming and extensions," in Linear programming and extensions: Princeton university press, 2016.

[22] M. H. Abdelati, M. I. Khalil, K. Abdelgawwad, and M. Rabie, "ALTERNATIVE ALGORITHMS FOR SOLVING CLASSICAL TRANSPORTATION PROBLEMS," Journal of Advanced Engineering Trends, vol. 39, no. 1, pp. 13-24, 2020.

[23] M. Marwan, "OPTIMASI BIAYA DISTRIBUSI MATERIAL DENGAN METODE NWC (NORTH WEST CORNER) DAN METODE VAM (VOGEL APPROXIMATION METHOD) PADA PT. XYZ," IESM Journal (Industrial Engineering System and Management Journal), vol. 2, no. 2, pp. 137-146, 2022.

[24] T. C. Purba, N. Notiragayu, F. Fitriani, and D. Azis, "The comparison of the effectiveness of the lowest supply lowest cost (LSLC) algorithm and the exponential approach algorithm in transportation problems," Desimal: Jurnal Matematika, vol. 4, no. 2, pp. 185-192, 2021.

[25] H. Dan, "Research on the Problems of Enterprise Logistics Transportation Cost Management and Optimization Countermeasures," in SHS Web of Conferences, 2022, vol. 148: EDP Sciences, p. 02006.

[26] M. H. Abdelati, A. M. Abd-El-Tawwab, E. E. M. Ellimony, and M. Rabie, "Solving a multi-objective solid transportation problem: a comparative study of alternative methods for decision-making," Journal of Engineering and Applied Science, vol. 70, no. 1, pp. 1-16, 2023.

[27] E. Ekanayake, W. B. Daundasekara, and S. P. C. Perera, "An Examination of Different Types of Transportation Problems and Mathematical Models," American Journal of Mathematical and Computer Modelling, vol. 7, no. 3,

pp. 37-48, 2022.
[28] D. M. Tulett and G. Y. Ke, "Using LINGO for Business Students," in Operations Research Forum, 2022, vol. 4, no. 1: Springer, p. 3.
[29] N. Gupta and I. Ali, Optimization with LINGO-18: Problems and Applications. CRC Press, 2021.