Verification and Validation Report: Minimization Analysis

Ning Wang

April 12, 2023

1 Revision History

Date	Version	Notes
Apr 12 2023	1.0	First Draft

Contents

1 Revision History **i**

2 Symbols, Abbreviations and Acronyms **1**

3 Implementation Validation **2**

3.1 Code Walkthrough 2

3.2 Functional Requirements Evaluation 3

3.3 Tests for Nonfunctional Requirements 6

3.3.1 Test NFR1: Accuracy 6

3.3.2 Test NFR2: Usability 6

3.3.3 NFR3: Maintainability 6

3.3.4 Test NFR4: Portability 6

4 Unit Tests **7**

4.1 Tests for Functional Requirements 7

4.1.1 Module 1: Test valid data 7

4.1.2 Module 2: Test invalid data 7

5 Changes Due to Testing **8**

6 Automated Testing **8**

7 Trace to Requirements **8**

8 Trace to Modules **8**

List of Tables

2 Symbols, Abbreviations and Acronyms

symbol	description
T	Test
VnV	verification and validation
Minimization Analysis	Core Analysis

The complete table of symbols, abbreviations and acronyms can be found in the SRS document of the software.

This document provides the information on validation and verification plans implemented for the Minimization Analysis software. In this regard, the general approaches and plans are initially discussed and afterwards specific test cases and approaches for validation and verification of functional and nonfunctional requirements (can be found in SRS) are reviewed. VnV plans here are a combination of manual (assigned to a member of the VnV team to assess) and automated testing approaches to evaluate the correctness of the information (whether input or output) or satisfaction of a goal in Minimization Analysis.

3 Implementation Validation

3.1 Code Walkthrough

This MATLAB program is designed to read data from a CSV file and optimize the power consumption and cost for a set of data centers using renewable and grid energy. The code reads information from the CSV file, extracts relevant data, sets up a linear programming optimization problem, and solves it using the linprog function. Here is a step-by-step walkthrough of the program:

- Clear the workspace and command window.
- Request the user to input the CSV filename.
- Read the CSV file into a table using the readtable function.
- Extract the number of data centers (numdatacenters) and other relevant information such as distances, rates of renewable and grid energy, total and maximum power consumption from the table.
- Display the extracted information.
- Define constants and set up matrices for the linear programming optimization problem. The objective function 'f' is defined as a combination of renewable and grid energy costs. The constraint matrices A, b, Aeq, and beq are set up to represent the restrictions on power consumption from renewable and grid sources.
- Check for NaN values in the matrices. If any are found, display an error message and exit the program.
- Solve the linear program using the linprog function.
- Check the solution exit flag. If the flag is not 1, display an error message and exit the program.
- Calculate the optimal power consumption for each data center from renewable and grid sources.
- Calculate the total cost of power consumption.

- Print the optimal power consumption and total cost for each data center.
- Plot the power consumption as a function of distance.
- Check if the constraints are satisfied. If not, print a message indicating that the constraints are not satisfied.
- Print the results for the current file and the total cost. If the constraints are satisfied, print "All constraints are satisfied." Otherwise, print "Constraints are not satisfied."

This program optimizes power consumption for a set of data centers, given the input data in a CSV file. It calculates the most cost-effective combination of renewable and grid energy to satisfy the power requirements while meeting the given constraints.

3.2 Functional Requirements Evaluation

Test for R1:

- R1: Requirements for the inputs that are supplied by the user. This information has to be explicit.
- The program requests the user to input the CSV filename, which contains the necessary data for the optimization problem. The user should be provided with clear instructions on how to format the CSV file, including the required columns and their order.

Test for R2:

- R2: The program shall notify the user if the input is out of bounds.
- The program checks for NaN values in the matrices but does not explicitly check for out-of-bound values in the input. Additional input validation should be implemented to notify the user if any input values are out of bounds.

```
% Clear workspace and command window
% Define the folder containing the CSV files
%csv_folder = 'path/to/your/csv/folder';

% Get a list of all CSV files in the folder
%csv_files = dir(fullfile(csv_folder, '*.csv'));

% Loop through all CSV files
%for file_idx = 1:length(csv_files)
   % filename = fullfile(csv_folder, csv_files(file_idx).name);
filename = input("Please enter the CSV filename: ", 's');
    % Read the CSV file
    T = readtable(filename);
    num_data_centers = size(T,1);

    % Extract Grid_Power_restrictions and Renewable_limits from the CSV file
    Grid_Power_restrictions = table2array(T(:, 7));
    Renewable_limits = table2array(T(:, 8));
```

Figure 1: System Context

Test for R3:

- R3: User should keep input validate data type.
- The program reads data from the CSV file and directly converts it into the required data types (e.g., arrays, scalars). However, there is no explicit validation of data types in the input. Additional input validation should be implemented to ensure that the user-supplied data is of the correct data type.

Test for R4:

- R4: The program should be able to convert units while minimizing cost.
- The program currently minimizes the cost of power consumption based on the given rates for renewable and grid energy. It does not involve unit conversion. If unit conversion is required, the program should be updated to handle different units and perform the necessary conversions while minimizing cost.

Test for R5:

- R5: The program should keep all decision variables under their constraints.
- The program sets up the constraint matrices (A, b, Aeq, and beq) and solves the linear programming optimization problem. After solving

the problem, it checks whether the constraints are satisfied within a specified tolerance. The program informs the user if the constraints are not satisfied.

Test for R6:

- R6: The program output the optimal plan for end users.
- The program prints the optimal power consumption for each data center, both from renewable and grid sources, and the total cost. It also plots the power consumption as a function of distance. The output provides the user with an optimal plan for power consumption and cost.

```
Number of data centers: 3
    "Distances to data centers: 100km"    "Distances to data centers: 200km"    "Distances to data centers: 300km"

Rate of renewable energy: 0.045
Rate of grid energy: 0.012
Total power consumption: 15
Maximum power consumption per data center: 10
Optimal power consumption for each data center:
Data Center 1:
Renewable energy: 0.09
Grid energy: 5
Actual Grid energy: 4.985
Data Center 2:
Renewable energy: 0
Grid energy: 5
Actual Grid energy: 4.97
Data Center 3:
Renewable energy: 0
Grid energy: 5
Actual Grid energy: 4.955
Total cost: 0.18297
Results for file: ex11
Total cost: 0.18297
All constraints are satisfied.
```

Figure 2: System Context

Test for R7:

- R7: The output should allocate the computing power of data centers with their capable size.
- The program optimizes power consumption based on the given constraints, such as maximum power consumption per data center. However, it does not explicitly allocate computing power. If allocation of computing power is required, the program should be updated to include this functionality.

3.3 Tests for Nonfunctional Requirements

3.3.1 Test NFR1: Accuracy

The accuracy of the computed solutions should meet the level needed for engineering application.

- The program uses the linprog function from MATLAB to solve the linear programming optimization problem, which is a well-established method for solving such problems. The accuracy of the computed solution should be sufficient for engineering applications. However, the user should validate the results and ensure the chosen tolerance level meets their specific requirements.

3.3.2 Test NFR2: Usability

The program shall not have a user interface but will clearly show the output, and data will be easy to copy and read.

- The program does not have a user interface, and it outputs the results in a clear and easy-to-read format. It prints the optimal power consumption and total cost for each data center and plots the power consumption as a function of distance. The user can easily copy the results and use them for further analysis.

3.3.3 NFR3: Maintainability

The time complexity of this program should be O(n).

- The program's time complexity is primarily determined by the linprog function, which has a time complexity that depends on the specific algorithm used and the problem size. Generally, the time complexity of linear programming solvers is higher than O(n). However, the program's overall complexity is dominated by the linear programming solver, and other operations in the program have relatively low time complexity.

3.3.4 Test NFR4: Portability

The program should be easily integrated with another software program.

- The program is written in MATLAB, which is widely used in engineering and scientific applications. It should be relatively easy to integrate the program with other MATLAB-based software. To integrate the program with software in other programming languages, the user may need to rewrite the program or use tools that enable communication between different programming languages (e.g., Python-MATLAB bridges or APIs).

4 Unit Tests

4.1 Tests for Functional Requirements

To run the unit test for this program, we'll refactor the program into 5 functions that can be individually tested. The detail of the unit test was illustrated in the command of code. The code can be found in Code

4.1.1 Module 1: Test valid data

4.1.2 Module 2: Test invalid data

5 Changes Due to Testing

NA

6 Automated Testing

The program finally achieves automated and the Automated Testing code can be found in Code

7 Trace to Requirements

All requirements are satisfied in section 3.

8 Trace to Modules

All requirements are satisfied in section 4.