Project Title: System Verification and Validation Plan for Minimization Analysis

Ning Wang

May 4, 2023

1 Revision History

Date	Version	Notes
Feb 8, 2023	1.0	First Draft
May 2, 2023	1.1	Revision 1

Contents

List of Tables

2 Symbols, Abbreviations and Acronyms

symbol	description
FR	Functional Requirement
MG	Module Guide
MIS	Module Interface Specification
NFR	Non-Functional Requirement
RSSC	Sadio SignalStrength Calculator
SRS	Software Requirement Specification
T	Test
VnV	Validation and Verification

This document outlines and details the verification and validation strategy for the minimization analysis program, aiming to confirm its compliance with the specified requirements. Section 3 offers a brief overview of the minimization analysis background, while Section 3.1 elaborates on the verification plan. In Section 5, the system testing procedures are discussed, encompassing tests for both functional and non-functional requirements.

3 General Information

3.1 Summary

The test for software written in MATLAB. The purpose of minimization analysis is to simulate the minimized cost and relocate the computational power for each data center. The inputs are defined by the user and find the analytical distribution of computational power which refer by load in this program for the user.

3.2 Objectives

The objective of this document is to build confidence in the software's level of correctness. To reach this objective, all functional and non-functional requirements will be tested following the descriptions in this document.

3.3 Relevant Documentation

- SRS

4 Verification and Validation Plan

Verification and Validation of minimization analysis include automated testing at the module level, the system level, and the integration level. This document will additionally propose continuous integration.

4.1 Verification and Validation Team

- Author: Ning Wang
- Primary Reviewer: Maryam Valian
- VnV Plan Reviewer: Joachim deFourestier
- MG +MIS Reviewer: Jason Balaci
- Dr. Spencer Smith
- class CAS 741

4.2 SRS Verification Plan

SRS will be done by team members reviewing the document. Team members can put any comments, suggestions or questions in Project's Github repository as issues. The author will respond to the issues and make modifications when needed.

4.3 Design Verification Plan

Design verification will be done by team members, by reviewing whether the steps of calculation in the software follows the physical model in SRS or not.

4.4 Implementation Verification Plan

Implementation verification will be done by testing all the functional and non-functional requirements. Descriptions of the tests can be found in subsection 5.1 and subsection 5.2. In addition, we will undergo automatic verification by checking all the codes built with MATLAB. The author will also conduct unit testing for modules within the testing scope. Details for unit testing can be found in VnV Report.

4.5 Automated Testing and Verification Tools

- MATLAB

4.6 Software Validation Plan

Validation is the validation of the requirements. Validation compares experimental data to output from minimization analysis to confirm or reject the problem model. Validation considers the applicability of the equations and assumptions to the problem space.

5 System Test Description

5.1 Tests for Functional Requirements

Functional requirements for this project are given in SRS section 5.1. There are 7 functional requirements for the minimization analysis model, from R1

to R7. R1, R2 and R3 are corresponding to inputs, while R6 and R7 are corresponding to outputs. subsubsection 5.1.1 describes the input tests for R1, R2 and R3; and subsubsection 5.1.3 describes the output tests for R6 and R7.

5.1.1 Input requirements

This test verifies the following requirements:

R1: Requirements for the inputs that are supplied by the user. This information has to be explicit.

R2: The program shall notify the user if the input is out of bounds.

R3: User should keep input validate data type.

5.1.2 Area of Testing1 - Fun. Req. 1 .2 and Fun. Req. 3 - Inputs

Functional Requirement 1 states The inputs that are supplied by the user. This information has to be explicit.

Functional Requirement 2 states: The program shall notify the user if an input value is illegal or out of bounds.

This area of tests will address the inputs to the program. The following tests include tests with inputs that are within bounds to test Fun. Req. 1 and tests with inputs that are out of bounds to test Fun. Req. 2.

Tests for Fun. Req. 1, 2 and 3

1. test - Normal Input using defaults

   Control: Automatic

   Initial State: Pending input

   Input: various distances di, using default inputs boundaries

   Output: The program should run with no error, (the correctness of the solution is addressed in other tests)

   Test Case Derivation: Fun. Req. 1, table 1

   How the test will be performed: A set of distances like in table 1, they are all typically normal numbers, the system will perform under default constrain and the program will run and output with the minimize plan.

2. test - Normal Input with user entered pricing

   Control: Automatic

   Initial State: Pending input

   Input: A set of distances and different pricing as default,

   Output: The program should run with no error, (the correctness of the solution is addressed in other tests)

   Test Case Derivation: Fun. Req. 1

   How the test will be performed: The system should use the input pricing to calculate the minimization value and output the distributed value. This time will conduct a set of two tests, one with default pricing, and another with tester-entered pricing, these testing will be with the same input distances. Tester will compare both results to observe the pricing unit affects the distribution of computational power for data centers.

3. test - Out of Bounds distances

   Control: Manual

   Initial State: Pending input

   Input: distances which are out of the distances limit. (Table 2)

   Output: The program should return an error.

   Test Case Derivation: Fun. Req. 2

   How test will be performed: The program will set the boundary for input, in this case, some distance out of the boundary will input like in table 2, the tester will expect to see the data center within the boundary will be assigned computational power, those out of boundary will return an error.

4. test- Out of Bounds for not reaching target consumption

   Control: Manual

   Initial State: Pending input

   Input: too less data centers to allocate target computational power

Output: The program should return an error about the analysis range

Test Case Derivation: Fun. Req. 2

How the test will be performed: In this test will assume up bound for the power of each data center that is capable. Tester will assign each data center with exceed the power and expect the program will return an error.

5. test- wrong data type
   Control: Manual

   Initial State: Pending input

   Input: input distances with the wrong data type

   Output: The program should return a warning and stop running.

   Test Case Derivation: Fun. Req. 3

   How the test will be performed: Tester will try a different type of data as input and expect to see the program stop running.

data center	distance(km)	total power consumption(MW)
1	1	100
2	5	
3	50	
4	500	
5	2000	

Table 1: Sample number of input for input test one

data center	distance(km)	total power consumption(MW)
1	1	100
2	5	
3	50	
4	5000	
5	20000	

Table 2: Sample number of input which some of them out of boundary

5.1.3 Output

This test verifies the following requirements:

R6: The program output the optimal plan for end users.

R7: The output should allocate the computing power of data centers with their capable size(1)

Output Test

1. Simple number of data centers valid output

   Control: Manual

   Initial State: Pending Input

   Input: Data centers number which exceeds needed.

   Output: Some data centers will be distributed, and some will return null

   Test Case Derivation: program returns the correct output.

   How the test will be performed: Tester will assign power to exceed data centers some data centers will be useless as in table 3. It is expected to see some data centers be fully assigned and some are assigned to null.

2. Simple multiple valid output

   Control: Manual

   Initial State: Pending Input

Input: Data centers with various that have a huge gap with each other. (1km, 5km, 50km, 500km, 2000km)

Output: In this case, it is expected to see the near data centers be assigned more computational power and the far one the less.

Test Case Derivation: The program returns the correct output.

How the test will be performed: Tester manually changes the inputs to see the changes in output.

data center	distance(km)	total power consumption(MW)
1	1	50
2	5	
3	20	
4	100	
5	200.3	
6	1000	
7	2000.5	
8	5000.1	

Table 3: Sample number of input which some of data center will not be assigned power

5.2 Tests for Non-Functional Requirements

Non-functional requirements are given in SRS section 5.2. There are 4 non-functional requirements: Accuracy, Usability, Maintainability, and Portability. The rest of this section provides detailed descriptions on how to test them.

5.2.1 Portability

Portability Test The program shall be able to run on different OS. Since the core algorithm is written on MATLAB, the program should fit every type of OS that can run MATLAB.[(2)]

1. Portability on Windows 10

   Type: Manual

   Initial State: The program is installed on a Windows 10 system with MATLAB.

   Input: Set of sample distances.

   Output: Execute the program on the Windows 10 system, input the sample distances, and verify that the program runs without errors and generates the expected output file.

   How the test will be performed: Pass

2. Portability on Linux Ubuntu

   Type: Manual

   Initial State: The program is installed on a Linux Ubuntu system with MATLAB.

   Input: Set of sample distances.

   Output: Execute the program on the Linux Ubuntu system, input the sample distances, and verify that the program runs without errors and generates the expected output file.

   How the test will be performed: Pass

5.2.2 Maintainable

Maintainability Test Proper documents should be included in this project.

1. Maintainability Test

   Type: Manual

   Initial State: none

   Input: none

   Output: none

   How the test will be performed: Tester manually checks the contents in the Github repo. On success, documents shall be uploaded following

the schedule of CAS741 and no issue shall be closed without a proper response.

5.2.3 Usability

Usability Test The program shall not have a user interface but will clearly show the output and data will easy to copy

1. Usability Test

   Type: Manual

   Initial State: none

   Input: none

   Output: none

   How to test will be performed: Tester reviews the code and since it is designed to write the result in excel, the tester will easy to copy and use the result.

5.2.4 Accuracy

Accuracy Test The accuracy of the computed solutions should meet the level needed for the engineering app.

1. Accuracy Test

   Type: Manual

   Initial State: Pending Input

   Input: A set of distances of data centers with power generation of known distributed computational power will be set up and tested the accuracy.

   Output: Tester will observe if the output matches the known result.

   How the test will be performed: Tester reviews the code and since we trust the library of MATLAB, the center algorithm of minimization analysis will be correct by default. The known result will allow 3% - 5% correctness.[3]

5.3 Traceability Between Test Cases and Requirements

	R1	R2	R3	R4	R5	R6	Portable	Maintainable
Input requirements	X	X						
Output			X	X	X			
Portability						X		
Maintainable							X	
Accuracy								X

Table 4: Traceability Between Test Cases and Requirements

6 Unit Test Description

6.1 Unit Testing Scope

This section is intentionally left blank until the MIS is completed.

References

[1] Richard P Sedano and Matthew H Brown. Electricity transmission: a primer. *National Council on Electricity Policy, June, Washington, DC*, 2004.

[2] Juan Rosellón. Different approaches towards electricity transmission expansion. *Review of network economics*, 2(3), 2003.

[3] Michel Rivier, Ignacio J Pérez-Arriaga, and Luis Olmos. Electricity transmission. *Regulation of the power sector*, pages 251–340, 2013.