Unit Verification and Validation Plan for a Library of Simplex Method Solvers (LoSMS)

Hanane Zlitni

December 3, 2018

1 Revision History

Date	Version	Notes
December 17, 2018	2.0	Final Draft (includes making the document consistent with the rest of the deliverables)
December 16, 2018	1.2	Applied Dr. Smith's Comments
December 6, 2018	1.1	Applied Brooks MacLachlan's Comments Posted on GitHub
December 3, 2018	1.0	First Draft

Contents

List of Tables

2 Symbols, Abbreviations and Acronyms

The following are symbols, abbreviations or acronyms used in this document:

symbol	**description**
T	Test
Z	Optimal solution(s) of the objective function
MIS	Module Interface Specification
V&V	Verification and Validation
LCs	Linear Constraints
ObjcFunc	Objective Function
Const	Constants
max	Maximization
min	Minimization

This document describes the unit Verification and Validation (V&V) plan for the Library of Simplex Method Solvers (LoSMS) tool. It is intended to be a refinement of the tool's system V&V plan by providing test cases based on the modules in the library's Module Interface Specification (MIS) document. The MIS, along with the full documentation of LoSMS, can be found at: https://github.com/hananezlitni/HZ-CAS741-Project.

The unit V&V plan starts by providing general information about the tool in Section 3. Then, Section 4 provides additional details about the plan, which include information about the V&V team, automated testing and verification tools and non-testing based verification. This is followed by the unit test description in Section 5, which consists of tests for the library's functional and nonfunctional requirements, categorized based on the modules in the MIS, and traceability between the test cases and modules.

3 General Information

3.1 Purpose

The software under test, LoSMS, is a general-purpose program family that is intended to be used by people from various backgrounds. It facilitates obtaining the optimal solution of a linear program using the simplex method. It accepts the objective function, the objective function goal (maximization or minimization) and the linear constraints and outputs the optimum of the linear programming problem.

3.2 Scope

The core module of the library is the simplex solver module. It contains all components of the tool from reading the inputs to calculating and obtaining the output. Specifically, the function *solveLP()* is the one that will be the focus of unit testing because all other functions are going to be called during its execution. Therefore, the test cases in this document, and the System V&V Plan (found at: https://github.com/hananezlitni/HZ-CAS741-Project/blob/master/docs/VnVPlan/SystVnVPlan/SystVnVPlan.pdf), are selected to cover all possible cases that could happen to *solveLP()*.

Some test cases are selected to cover the cases of the occurence of exceptions. These will help verify the exceptions module along with the solver module.

More information about the modules can be found in the MG document at: https://github.com/hananezlitni/HZ-CAS741-Project/blob/master/docs/Design/MG/MG.pdf.

[I suggest you give the specific names of the modules that will be verified. A reference to the MG would also be a good idea. —SS][done —HZ]

4 Plan

4.1 Verification and Validation Team

The verification and validation team consists of one member: Hanane Zlitni.

4.2 Automated Testing and Verification Tools

Unittest, a unit testing framework for Python, will be used for automated testing. Code coverage would be verified using Coverage.py.

4.3 Non-Testing Based Verification

Not applicable for LoSMS.

5 Unit Test Description

The test cases discussed in this section were selected to ensure that the library does correctly what it is intended to do while maintaining quality. The test cases were derived from the tool's MIS document which can be found at: https://github.com/hananezlitni/HZ-CAS741-Project/blob/master/docs/Design/MIS/MIS.pdf

5.1 Tests for Functional Requirements

The following are the test cases related to the tool's functional requirements and categorized based on the modules in the MIS document.

5.1.1 Simplex Solver Module

The simplex solver module is the main module of the library. It receives the inputs from the driver program and performs the calculations necessary to obtain the optimal solution of the linear program.

Testing for faulty inputs is detailed in the System V&V Plan (T5 & T6) found at: https://github.com/hananezlitni/HZ-CAS741-Project/blob/master/docs/VnVPlan/SystVnVPlan/SystVnVPlan.pdf

As previously explained in Section 3.2, the focus of the test cases in this document will be on *solveLP()*. The following test cases were selected to test each possible case that can occur in the function execution.

1. **T1: First Test for solveLP() for maximization problems**

   Control: Automatic

   Initial State: -

   Input: LCs = [[20,6,3],[0,1,0],[-1,-1,1],[-9,1,1]], Constants = [182,10,0,0], ObjcFunc = [100000,40000,18000], Goal = max

   Output: $Z = 1052000$, $x_1 = 4$, $x_2 = 10$, $x_3 = 14$

   Test Case Derivation: Section 6 in the MIS (Zlitni (2018))

   How test will be performed: Unittest

2. **T2: Second Test for solveLP() for maximization problems**

   Control: Automatic

   Initial State: -

   Input: LCs = [[3,2],[1,0],[0,1],[-9,1,1]], Constants = [182,40,60], ObjcFunc = [3,2], Goal = max

   Output: $Z = 180$, $x_1 = 40$, $x_2 = 30$

   Test Case Derivation: Section 6 in the MIS (Zlitni (2018))

   How test will be performed: Unittest

3. **T3: Test solveLP() for minimization problem with no optimal solution**

   Control: Automatic

Initial State: -

Input: LCs = [[400,300],[300,400],[200,500]], Constants = [25000,27000,30000], ObjcFunc = [11,16,15], Goal = min

Output: Exception message: "This linear program does not have an optimal solution"

Test Case Derivation: Section 6 in the MIS (Zlitni (2018))

How test will be performed: Unittest

[I like your explanations; they add to the document. Unfortunately they will disappear when the comments are turned off. I suggest that you move this information out of the comment and into the document. You might even keep the heading explanation and add it to your template. —SS]

5.1.2 Explanation

The function *solveLP()* receives 4 inputs in the following order:

1. The coefficients of the linear constraints (LCs) in the form of a 2-dimentional array
2. The constants (const) in the linear constraints, such that: Ax $\leq$ const
3. The coefficients of the objective function (ObjcFunc)
4. The linear program goal (max or min)

Most of the test cases for the library are system tests (detailed in the System V&V Plan). The test cases in this document are added to ensure a high test coverage.

5.2 Tests for Nonfunctional Requirements

The test cases T7-T12 in section 5.2 in the System V&V Plan cover the non-functional requirements (qualities) that are important for LoSMS. The document can be found at: https://github.com/hananezlitni/HZ-CAS741-Project/blob/master/docs/VnVPlan/SystVnVPlan/SystVnVPlan.pdf.

5.3 Code Coverage

[Sorry for changing the template.. but I wanted to add a section for code coverage because it is planned for LoSMS. —HZ]

It is planned to verify the code coverage that the test cases for LoSMS achieves. The aim is to reach a minimum of 99% code coverage. In the case of a lower percentage, more test cases would be added for the parts of the implementation that weren't covered.

As explained at the beginning of this document, Python's Coverage.py will be used to verify the code coverage. It provides a table with the coverage percentage along with a detailed view on the parts of the implementation that were and weren't covered.

5.4 Traceability Between Test Cases and Modules

Test Case Number	Module
T5 & T6 in System V&V Plan	Simplex Solver Module
T1	Simplex Method Solver Module
T2	Simplex Method Solver Module
T3	Simplex Solver Module, Exceptions Module

Table 1: Traceability Between Test Cases and Modules

[test cases look good. —SS]

References

Hanane Zlitni. Module interface specification for a library of simplex method solvers (losms), 2018. URL https://github.com/hananezlitni/HZ-CAS741-Project/blob/master/docs/Design/MIS/MIS.pdf.

6 Appendix

This section provides additional content related to this document.

6.1 Symbolic Parameters

There are no symbolic parameters used in this document.