Module Interface Specification for Library of Linear Algebraic Equation Solver

Devi Prasad Reddy Guttapati

November 30, 2017

1 Revision History

Date	Version	Notes
Date 1	1.0	Initial Draft

2 Symbols, Abbreviations and Acronyms

See SRS Documentation at: https://github.com/deviprasad135/CAS741/blob/master/Doc/SRS/CA.pdf

Contents

3 Introduction

The following document details the Module Interface Specifications for Library of Linear Algebraic Equation Solver.

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at the mentioned link below:

https://github.com/deviprasad135/CAS741.

4 Notation

The structure of the MIS for modules comes from Hoffman and Strooper (1999), with the addition that template modules have been adapted from Ghezzi et al. (2002). The mathematical notation comes from Chapter 3 of Hoffman and Strooper (1999). For instance, the symbol := is used for a multiple assignment statement and conditional rules follow the form $(c_1 \Rightarrow r_1 | c_2 \Rightarrow r_2 | ... | c_n \Rightarrow r_n)$.

The following table summarizes the primitive data types used by Library of Linear Algebraic Equation Solver.

Data Type	**Notation**	**Description**
character	char	a single symbol or digit
integer	$\mathbb{Z}$	a number without a fractional component in (-∞, ∞)
natural number	$\mathbb{N}$	a number without a fractional component in [1, ∞)
real	$\mathbb{R}$	any number in (-∞, ∞)

The specification of Library of Linear Algebraic Equation Solver uses some derived data types: sequences, strings, and tuples. Sequences are lists filled with elements of the same data type. Strings are sequences of characters. Tuples contain a list of values, potentially of different types. In addition, Library of Linear Algebraic Equation Solver uses functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification.

5 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

Level 1	**Level 2**
Hardware-Hiding	
Behaviour-Hiding	Input Computing Module Output computing Module Library of Linear Algebraic Equation Solver Module
Software Decision	Matrix Module Gaussian Elimination Module Gauss-Jordan Elimination Module

Table 1: Module Hierarchy

6 MIS of Library of Linear Algebraic Equation Solver Module

6.1 Module

LLAES

6.2 Uses

IC (Section 7), OC (Section 8), Matrix (Section 9), GE (Section 10), GJE (Section 11)

6.3 Syntax

Name	**In**	**Out**	**Exceptions**
Linear_Algebraic Equation_Solving Methods	Linear_Algebraic Equation_Solving Method $\in 1, 2$	-	-

6.4 Semantics

6.4.1 State Variables

None

6.4.2 Access Routine Semantics

- transition: None
- output: None
- exception: None
- pseudocode:

```
if (option = 1)
   then solve using Gaussian Elimination Method
if (option = 2)
   then solve using Gauss-Jordan Elimination Method
```

7 MIS of Input Computing Module

7.1 Module

IC

7.2 Uses

Matrix (Section 9), GE (Section 10), GJE (Section 11)

7.3 Syntax

Name	In	Out	Exceptions
A	$n \times n$ matrix $\in \mathbb{R}$ and n > 0	-	Complex Numbers
b	$n \times 1$ matrix $\in \mathbb{R}$ and n > 0	-	Complex Numbers

7.4 Semantics

7.4.1 State Variables

None

7.4.2 Access Routine Semantics

- transition: None
- output: None
- exception: None

8 MIS of Output Computing Module

8.1 Module

OC

8.2 Uses

None

8.3 Syntax

Name	In	Out	Exceptions
x	-	$n \times 1$ matrix $\in \mathbb{R}$ and n > 0	-

8.4 Semantics

8.4.1 State Variables

None

8.4.2 Access Routine Semantics

- transition: None
- output: None
- exception: None
- pseudocode:

```
if (displayresult)
    {print(x);}
end if
```

9 MIS of Matrix Module

9.1 Module

Matrix

9.2 Uses

GE (Section 10), GJE (Section 11), OC (Section 8)

9.3 Syntax

Name	In	Out	Exceptions
A	$\mathbb{R}$	$n \times n$ matrix $\in \mathbb{R}$ and n > 0	-
b	$\mathbb{R}$	$n \times 1$ matrix $\in \mathbb{R}$ and n > 0	-

9.4 Semantics

9.4.1 State Variables

None

9.4.2 Access Routine Semantics

- transition: None
- output: None
- exception: None
- pseudocode:

```
function{
            A = (Input of Real Numbers)
            n = $(length (A))^{1/2}$
            A = matrix(A, row = n, col = n, byrow = TRUE)
            }
```

10 MIS of Gaussian Elimination Module

10.1 Module

GE

10.2 Uses

This module is used to solve the system of Linear Algebraic Equations.

10.3 Syntax

Name	In	Out	Exceptions
A	$n \times n$ matrix $\in \mathbb{R}$ and n > 0	-	always_a_square_matrix no_singular_matrix
b	$n \times 1$ matrix $\in \mathbb{R}$ and n > 0	-	-
x	-	$n \times 1$ matrix $\in \mathbb{R}$ and n > 0	-

10.4 Semantics

10.4.1 State Variables

None

10.4.2 Access Routine Semantics

- transition: None
- output: None
- exception: None
- pseudocode:

```
for k = 1 to n − 1
    find a pivot p such that
    |a_pk| ≥ |a_ik| for K ≤ i ≤ n
    if |a_pk| = 0 do
        return "Singular Matrix"
        end the entire loop
    else
        interchange row p and k
```

```
    for i = k+1 to n
        factor_ik = a_ik / a_kk
        for j = k+1 to n
            a_ij = a_ij - factor_ik * a_kj
        end for
    end for
end for

x_n = b'_n / a_nn
for i in n-1 to 1
    for j in i+1 to n
        sum = a_ij x_j
    end for
    x_i = (b'_n - sum) / a_ii
end for
```

11 MIS of Gauss-Jordan Elimination Module

11.1 Module

GJE

11.2 Uses

This module is used to solve the system of Linear Algebraic Equations.

11.3 Syntax

Name	In	Out	Exceptions
A	$n \times n$ matrix $\in \mathbb{R}$ and n > 0	-	always_a_square_matrix no_singular_matrix
b	$n \times 1$ matrix $\in \mathbb{R}$ and n > 0	-	-
x	-	$n \times 1$ matrix $\in \mathbb{R}$ and n > 0	-

11.4 Semantics

11.4.1 State Variables

None

11.4.2 Access Routine Semantics

- transition: None
- output: None
- exception: None
- pseudocode:

```
for k = 1 to n − 1
    find a pivot p such that
    |a_pk| ≥ |a_ik| for K ≤ i ≤ n
    if |a_pk| = 0 do
        return "Singular Matrix"
        end the entire loop
    else
        interchange row p and k

    for i = k + 1 to n
        factor_ik = a_ik / a_kk
        for j = k + 1 to n
            a_ij = a_ij − factor_ik * a_kj
        end for
    end for
end for

Assuming that the matrix is not singular
for k = n to 2
    for i = k+1 to 1
        factor_ik = a_ik / a_kk
        for j = k−1 to 1
            a_ij = a_ij − factor_ik * a_kj
        end for
    end for
end for
```

```
for i in 1 to n
```

$$x_i = \frac{b_n^{'}}{a_{ii}}$$

```
end for
```

References

Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of software engineering*. Prentice Hall PTR, 2002.

Daniel Hoffman and Paul Strooper. Software design, automated testing, and maintenance–a practical approach. 1999.

12 Appendix

Not Applicable