

# Module Interface Specification for Time\_Freq\_Analysis

Elizabeth Hofer

December 4, 2020

# 1 Revision History

Date	Version	Notes
11.19.2020	1.0	Initial Release

## 2 Symbols, Abbreviations and Acronyms

See SRS Documentation at [https://github.com/liziscool/cas741\\_project/blob/master/docs/SRS/SRS.pdf](https://github.com/liziscool/cas741_project/blob/master/docs/SRS/SRS.pdf)

# Contents

<b>1</b>	<b>Revision History</b>	<b>i</b>
<b>2</b>	<b>Symbols, Abbreviations and Acronyms</b>	<b>ii</b>
<b>3</b>	<b>Introduction</b>	<b>1</b>
<b>4</b>	<b>Notation</b>	<b>1</b>
<b>5</b>	<b>Module Decomposition</b>	<b>1</b>
<b>6</b>	<b>MIS of Control Module</b>	<b>3</b>
6.1	Module . . . . .	3
6.2	Uses . . . . .	3
6.3	Syntax . . . . .	3
6.3.1	Exported Constants . . . . .	3
6.3.2	Exported Access Programs . . . . .	3
6.4	Semantics . . . . .	3
6.4.1	State Variables . . . . .	3
6.4.2	Environment Variables . . . . .	3
6.4.3	Assumptions . . . . .	3
6.4.4	Access Routine Semantics . . . . .	3
6.4.5	Local Functions . . . . .	4
<b>7</b>	<b>MIS of Specification Parameter Module</b>	<b>5</b>
7.1	Module . . . . .	5
7.2	Uses . . . . .	5
7.3	Syntax . . . . .	5
7.3.1	Exported Constants . . . . .	5
7.3.2	Exported Access Programs . . . . .	5
7.4	Semantics . . . . .	5
7.4.1	State Variables . . . . .	5
7.4.2	Environment Variables . . . . .	5
7.4.3	Assumptions . . . . .	5
7.4.4	Access Routine Semantics . . . . .	5
7.4.5	Local Functions . . . . .	6
7.5	Considerations . . . . .	6
<b>8</b>	<b>MIS of Input Param Module</b>	<b>7</b>
8.1	Module . . . . .	7
8.2	Uses . . . . .	7
8.3	Syntax . . . . .	7
8.3.1	Exported Constants . . . . .	7

8.3.2	Exported Access Programs . . . . .	7
8.4	Semantics . . . . .	7
8.4.1	State Variables . . . . .	7
8.4.2	Environment Variables . . . . .	8
8.4.3	Assumptions . . . . .	8
8.4.4	Access Routine Semantics . . . . .	8
8.4.5	Local Functions . . . . .	9
<b>9</b>	<b>MIS of Read Data Module</b>	<b>10</b>
9.1	Module . . . . .	10
9.2	Uses . . . . .	10
9.3	Syntax . . . . .	10
9.3.1	Exported Constants . . . . .	10
9.3.2	Exported Access Programs . . . . .	10
9.4	Semantics . . . . .	10
9.4.1	State Variables . . . . .	10
9.4.2	Environment Variables . . . . .	10
9.4.3	Assumptions . . . . .	10
9.4.4	Access Routine Semantics . . . . .	10
9.4.5	Local Functions . . . . .	11
<b>10</b>	<b>MIS of Boundary Configuration Module</b>	<b>11</b>
10.1	Module . . . . .	11
10.2	Uses . . . . .	11
10.3	Syntax . . . . .	11
10.3.1	Exported Constants . . . . .	11
10.3.2	Exported Access Programs . . . . .	11
10.4	Semantics . . . . .	11
10.4.1	State Variables . . . . .	11
10.4.2	Environment Variables . . . . .	11
10.4.3	Assumptions . . . . .	11
10.4.4	Access Routine Semantics . . . . .	12
10.4.5	Local Functions . . . . .	12
10.5	Considerations . . . . .	12
<b>11</b>	<b>MIS of STFT Module</b>	<b>12</b>
11.1	Module . . . . .	12
11.2	Uses . . . . .	12
11.3	Syntax . . . . .	12
11.3.1	Exported Constants . . . . .	12
11.3.2	Exported Access Programs . . . . .	12
11.4	Semantics . . . . .	13
11.4.1	State Variables . . . . .	13

11.4.2	Environment Variables . . . . .	13
11.4.3	Assumptions . . . . .	13
11.4.4	Access Routine Semantics . . . . .	13
11.4.5	Local Functions . . . . .	13
<b>12</b>	<b>MIS of Wavelet</b>	<b>13</b>
12.1	Module . . . . .	13
12.2	Uses . . . . .	13
12.3	Syntax . . . . .	14
12.3.1	Exported Constants . . . . .	14
12.3.2	Exported Access Programs . . . . .	14
12.4	Semantics . . . . .	14
12.4.1	State Variables . . . . .	14
12.4.2	Environment Variables . . . . .	14
12.4.3	Assumptions . . . . .	14
12.4.4	Access Routine Semantics . . . . .	14
12.4.5	Local Functions . . . . .	14
<b>13</b>	<b>MIS of Output Verification Module</b>	<b>15</b>
13.1	Module . . . . .	15
13.2	Uses . . . . .	15
13.3	Syntax . . . . .	15
13.3.1	Exported Constants . . . . .	15
13.3.2	Exported Access Programs . . . . .	15
13.4	Semantics . . . . .	15
13.4.1	State Variables . . . . .	15
13.4.2	Environment Variables . . . . .	15
13.4.3	Assumptions . . . . .	15
13.4.4	Access Routine Semantics . . . . .	15
13.4.5	Local Functions . . . . .	16
<b>14</b>	<b>MIS of Output Module</b>	<b>16</b>
14.1	Module . . . . .	16
14.2	Uses . . . . .	16
14.3	Syntax . . . . .	16
14.3.1	Exported Constants . . . . .	16
14.3.2	Exported Access Programs . . . . .	16
14.4	Semantics . . . . .	16
14.4.1	State Variables . . . . .	16
14.4.2	Environment Variables . . . . .	16
14.4.3	Assumptions . . . . .	16
14.4.4	Access Routine Semantics . . . . .	17
14.4.5	Local Functions . . . . .	17

<b>15 MIS of Compute Transform</b>	<b>17</b>
15.1 Module . . . . .	17
15.2 Uses . . . . .	17
15.3 Syntax . . . . .	17
15.3.1 Exported Constants . . . . .	17
15.3.2 Exported Access Programs . . . . .	17
15.4 Semantics . . . . .	17
15.4.1 State Variables . . . . .	17
15.4.2 Environment Variables . . . . .	17
15.4.3 Assumptions . . . . .	18
15.4.4 Access Routine Semantics . . . . .	18
15.4.5 Local Functions . . . . .	18
15.5 Considerations . . . . .	18
<b>16 MIS of Zero Pad Module</b>	<b>18</b>
16.1 Module . . . . .	18
16.2 Uses . . . . .	18
16.3 Syntax . . . . .	19
16.3.1 Exported Constants . . . . .	19
16.3.2 Exported Access Programs . . . . .	19
16.4 Semantics . . . . .	19
16.4.1 State Variables . . . . .	19
16.4.2 Environment Variables . . . . .	19
16.4.3 Assumptions . . . . .	19
16.4.4 Access Routine Semantics . . . . .	19
16.4.5 Local Functions . . . . .	19
<b>17 MIS of Matrix Data Structure Module</b>	<b>19</b>
17.1 Module . . . . .	19
17.2 Uses . . . . .	20
17.3 Syntax . . . . .	20
17.3.1 Exported Constants . . . . .	20
17.3.2 Exported Access Programs . . . . .	20
17.4 Semantics . . . . .	20
17.4.1 State Variables . . . . .	20
17.4.2 Environment Variables . . . . .	20
17.4.3 Assumptions . . . . .	20
17.4.4 Access Routine Semantics . . . . .	20
17.4.5 Local Functions . . . . .	20

<b>18 MIS of Plotting Module</b>	<b>21</b>
18.1 Module . . . . .	21
18.2 Uses . . . . .	21
18.3 Syntax . . . . .	21
18.3.1 Exported Constants . . . . .	21
18.3.2 Exported Access Programs . . . . .	21
18.4 Semantics . . . . .	21
18.4.1 State Variables . . . . .	21
18.4.2 Environment Variables . . . . .	21
18.4.3 Assumptions . . . . .	21
18.4.4 Access Routine Semantics . . . . .	21
18.4.5 Local Functions . . . . .	22
<b>19 Appendix</b>	<b>23</b>



### 3 Introduction

The following document details the Module Interface Specifications for Time\_Freq\_Analysis, a program to compute the time-frequency analysis of a 1 dimensional signal.

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at [https://github.com/liziscool/cas741\\_project](https://github.com/liziscool/cas741_project).

### 4 Notation

The structure of the MIS for modules comes from Hoffman and Strooper (1995), with the addition that template modules have been adapted from Ghezzi et al. (2003). The mathematical notation comes from Chapter 3 of Hoffman and Strooper (1995). For instance, the symbol  $:=$  is used for a multiple assignment statement and conditional rules follow the form  $(c_1 \Rightarrow r_1 | c_2 \Rightarrow r_2 | \dots | c_n \Rightarrow r_n)$ .

The following table summarizes the primitive data types used by Time\_Freq\_Analysis.

Data Type	Notation	Description
character	char	a single symbol or digit
integer	$\mathbb{Z}$	a number without a fractional component in $(-\infty, \infty)$
natural number	$\mathbb{N}$	a number without a fractional component in $[1, \infty)$
real	$\mathbb{R}$	any number in $(-\infty, \infty)$
complex	$\mathbb{C}$	a number with a real part $a$ and an imaginary part $b$ s.t. $a + bi$ where $i$ is the imaginary number

The specification of Time\_Freq\_Analysis 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, Time\_Freq\_Analysis 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 Module	
	Input Parameter Module
	Specification Param Module
	Read Data Module
Behaviour-Hiding Module	Boundary Configuration Module
	STFT Module
	Wavelet Module
	Output Verification Module
	Output Module
	Compute Transform Module
	Control Module
	Plotting Module
Software Decision Module	Zero-Pad Module
	Matrix Data Structure Module
	Fast Fourier Transform Module

Table 1: Module Hierarchy

## 6 MIS of Control Module

### 6.1 Module

main

### 6.2 Uses

Compute Transform Module 15, Output Module 14, Output Verification Module, Plotting Module 18

### 6.3 Syntax

#### 6.3.1 Exported Constants

None.

#### 6.3.2 Exported Access Programs

Name	In	Out	Exceptions
main	$argc : \mathbb{N},$ $argv_1, \dots, argv_{argc}$ s.t. $argv_n : string$	$< -$ $>$	-

### 6.4 Semantics

#### 6.4.1 State Variables

None.

#### 6.4.2 Environment Variables

None.

#### 6.4.3 Assumptions

- User enters correct inputs for calculation that is expected.
- User enters inputs in correct format.

#### 6.4.4 Access Routine Semantics

main():

- transition: Controls the entire program, transition through the program as follows:
  1. Sets `param` from command line arguments `set_inputs()`.

2. Calls compute transform module with `comp_transform()`.
3. Plots output with `plot_matrix()`

#### **6.4.5 Local Functions**

None.

## 7 MIS of Specification Parameter Module

### 7.1 Module

spec\_param

### 7.2 Uses

None.

### 7.3 Syntax

#### 7.3.1 Exported Constants

Name	Type
MIN_FREQ	$\mathbb{R}$
MAX_FREQ	$\mathbb{R}$
MIN_SIG_LEN	$\mathbb{N}$
MAX_SIG_LEN	$\mathbb{N}$
STEP_SIZE	$\mathbb{N}$
WIN_SIZE	$\mathbb{N}$

#### 7.3.2 Exported Access Programs

None.

### 7.4 Semantics

#### 7.4.1 State Variables

None.

#### 7.4.2 Environment Variables

None.

#### 7.4.3 Assumptions

None.

#### 7.4.4 Access Routine Semantics

None.

#### **7.4.5 Local Functions**

None.

### **7.5 Considerations**

This module basically just holds all of the constants.

## 8 MIS of Input Param Module

### 8.1 Module

input\_param

### 8.2 Uses

Specification Param Module 7, Hardware Hiding Module

### 8.3 Syntax

#### 8.3.1 Exported Constants

None.

#### 8.3.2 Exported Access Programs

Name	In	Out	Exceptions
set_input	$argc : \mathbb{N}, < -$ $argv_1, \dots, argv_{argc} >$ s.t. $argv_n :$ $string$	-	bad_arguments, bad_min_time, bad_max_time, bad_time_range, bad_min_freq, bad_max_freq, bad_freq_range , bad_transform_type
param.N	-	$\mathbb{N}$	
$f_1$	-	$\mathbb{R}$	
$f_2$	-	$\mathbb{R}$	
$n_1$	-	$\mathbb{R}$	
$n_2$	-	$\mathbb{R}$	
transform_type	-	$\{ 'W', 'S' \}$	
param.time_res	-	$\mathbb{N}$	
param.freq_res	-	$\mathbb{N}$	
param.sig_file	-	string	

### 8.4 Semantics

#### 8.4.1 State Variables

None.

### 8.4.2 Environment Variables

None.

### 8.4.3 Assumptions

While there will be measures in place to check that the input values comply with the ability of the program, some additional assumptions are listed below

1. The user inputs the correct file path.
2. The bounds are appropriate for the signal.
3. The input parameters must be set (initialized) before they are accessed.

### 8.4.4 Access Routine Semantics

set\_input():

- output: None.
- exception: exc:=

$\left\{ \begin{array}{l} \textit{bad\_arguments} \\ \textit{bad\_min\_time} \\ \textit{bad\_max\_time} \\ \textit{bad\_time\_range} \\ \textit{bad\_min\_freq} \\ \textit{bad\_max\_freq} \\ \textit{bad\_freq\_range} \\ \textit{bad\_transform\_type} \end{array} \right.$	$\left\{ \begin{array}{l} \text{if any of the arguments are incorrectly flagged (i.e. using a flag that doesn't exist)} \\ \text{if } n_1 \geq n_2 \\ \text{if } n_2 > N \\ \text{if } (n_2 - n_1) > MAX\_SIG\_LEN \text{ or } (n_2 - n_1) < MIN\_SIG\_LEN \\ \text{if } f_1 < MIN\_FREQ \\ \text{if } f_2 < MAX\_FREQ \\ \text{if } (f_2 - f_1) > MAX\_FREQ\_RANGE \text{ or } (f_2 - f_1) < MIN\_FREQ\_RANGE \\ \text{if T is not 'W' or 'S'} \end{array} \right.$
---	--

param.N :

- output:  $out := N$
- exception: none

param.f1 :

- output:  $out := f_1$
- exception: none

param.f2 :

- output:  $out := f_2$



- exception: none

param.n1 :

- output:  $out := n_1$

- exception: none

param.n2 :

- output:  $out := n_2$

- exception: none

param.time\_div :

- output:  $out := time\_div$

- exception: none

param.freq\_div :

- output:  $out := freq\_div$

- exception: none

param.time\_res :

- output:  $out := time\_res$

- exception: none

param.freq\_res :

- output:  $out := freq\_res$

- exception: none

param.sig\_filename :

- output:  $out := sig\_filename$

- exception: none

#### 8.4.5 Local Functions

None.

## 9 MIS of Read Data Module

### 9.1 Module

read\_data

### 9.2 Uses

Input Param Module 8

### 9.3 Syntax

#### 9.3.1 Exported Constants

None.

#### 9.3.2 Exported Access Programs

Name	In	Out	Exceptions
read_sig	sig_filename:string	$x_1, \dots, x_N > x_n : \mathbb{R}$	bad_path, empty_file, file_wrong_format

### 9.4 Semantics

#### 9.4.1 State Variables

None.

#### 9.4.2 Environment Variables

None.

#### 9.4.3 Assumptions

1. File should be in correct format.

#### 9.4.4 Access Routine Semantics

read\_sig():

- output:  $\text{out} := \langle x_1, \dots, x_N \rangle$

- exception:  $\text{exc} :=$

$$\begin{cases} \text{bad\_file} & \text{if the file or a directory on the path does not exist} \\ \text{empty\_file} & \text{if the file has no data in it} \\ \text{file\_format\_wrong} & \text{if data in the folder is not in } \mathbb{R} \text{ or uses incorrect delimiter} \end{cases}$$

#### 9.4.5 Local Functions

None.

## 10 MIS of Boundary Configuration Module

### 10.1 Module

bound\_config

### 10.2 Uses

Specification Param Module 7

### 10.3 Syntax

#### 10.3.1 Exported Constants

None.

#### 10.3.2 Exported Access Programs

Name	In	Out	Exceptions
calc_freq_res	$f_1 : \mathbb{R}, f_2 : \mathbb{R}, \text{win\_size} : \mathbb{N}$	$\mathbb{R}$	-
calc_time_res	$n_1 : \mathbb{N}, n_2 : \mathbb{R}, \text{step\_size} : \mathbb{N}$	$\mathbb{N}$	-

### 10.4 Semantics

#### 10.4.1 State Variables

None.

#### 10.4.2 Environment Variables

None.

#### 10.4.3 Assumptions

None.

#### 10.4.4 Access Routine Semantics

calc\_freq\_res():

- output:  $\text{out} := (f_2 - f_1) * \text{win\_size}$

calc\_time\_res():

- output:  $\text{out} := \frac{(n_2 - n_1)}{\frac{\text{win\_size}}{\text{step\_size}}}$

#### 10.4.5 Local Functions

None.

### 10.5 Considerations

At the time this document was written, the writer is not totally confident in the methods to calculate time\_res or freq\_res. The equations provided above are sufficient to communicate the point, but in execution it may be more complicated, and at that time this section of the document will be updated to reflect that.

## 11 MIS of STFT Module

### 11.1 Module

STFT

### 11.2 Uses

FFT Module, Boundary Configuration Module [10](#) Matrix Data Structure Module [17](#)

### 11.3 Syntax

#### 11.3.1 Exported Constants

None.

#### 11.3.2 Exported Access Programs

Name	In	Out	Exceptions
comp_stft	$\langle x_1, \dots, x_N \rangle$ where $x_n : \mathbb{R}, N : \mathbb{N}$ , time_res: $\mathbb{N}$ , freq_res: $\mathbb{N}$	$\langle X_{0,0}, X_{0,1}, \dots, X_{1,0}, X_{1,1}, \dots, X_{L,J} \rangle$ where $x_{i,j} : \mathbb{C}$	-

## 11.4 Semantics

### 11.4.1 State Variables

None.

### 11.4.2 Environment Variables

None.

### 11.4.3 Assumptions

None.

### 11.4.4 Access Routine Semantics

comp\_stft():

- output:  $\text{out} := \langle X_{0,0}, X_{0,1}, \dots, X_{1,0}, X_{1,1}, \dots, X_{I,J} \rangle$  where  $X_{i,j} : \mathbb{C}$  s.t.

$$X(i, j) = \sum_{i=0}^{WIN\_SIZE} x_i w_i e^{-\hat{i}\omega j} \quad (1)$$

and  $I[0, time\_res]$  and  $J[0, freq\_res]$  and  $\hat{i}$  is the imaginary number.

- This routine will utilize the fast Fourier transform.

### 11.4.5 Local Functions

window\_function():

- output:  $\text{out} := \langle w_0, \dots, w_{WIN\_SIZE} \rangle$  where

$$w_n = \left( \sin \frac{\pi * n}{WIN\_SIZE} \right)^2 \quad (2)$$

## 12 MIS of Wavelet

### 12.1 Module

wavelet

### 12.2 Uses

FFT Module, Boundrey Configuration Module [10](#), Matrix Data Structure Module [17](#)

## 12.3 Syntax

### 12.3.1 Exported Constants

None.

### 12.3.2 Exported Access Programs

Name	In	Out	Exceptions
comp_waveletT	$\langle x_1, \dots, x_N \rangle$ where $x_n : \mathbb{R}, N : \mathbb{N},$ time_res: $\mathbb{N}$ , freq_res : $\mathbb{N}$	$\langle$ $X_{0,0}, X_{0,1}, \dots, X_{1,0}, X_{1,1}, \dots, X_{I,J} \rangle$ where $x_{i,j} : \mathbb{C}$	-

## 12.4 Semantics

### 12.4.1 State Variables

None.

### 12.4.2 Environment Variables

None.

### 12.4.3 Assumptions

None.

### 12.4.4 Access Routine Semantics

comp\_waveletT():

- output: out :=  $\langle X_{0,0}, X_{0,1}, \dots, X_{1,0}, X_{1,1}, \dots, X_{I,J} \rangle$  s.t.

$$X(a, b) = \frac{1}{\sqrt{a}} \sum_{n=0}^{wav\_scale\_a} \Psi_{a,b,n} x_n \quad (3)$$

where  $\Psi_{a,b,n}$  represents the wavelet scaled by  $a$  and shifted by  $b$ .

### 12.4.5 Local Functions

wavelet\_function():

- output: out :=  $\langle w_0, \dots, w_{wav\_scale\_a} \rangle$  where

$$w_n = c_\sigma \pi^{-\frac{1}{4}} e^{-\frac{1}{2}t} (e^{i\sigma n} - \kappa_\sigma) \quad (4)$$

and where  $\kappa_\sigma = e^{\frac{1}{2}\sigma^2}$  and  $c_\sigma = (1 + e^{-\sigma^2} - 2e^{-\frac{3}{4}\sigma^2})^{\frac{1}{2}}$ , a.k.a. a Morlet Wavelet.

## 13 MIS of Output Verification Module

### 13.1 Module

output\_verify

### 13.2 Uses

Matrix Data Module

### 13.3 Syntax

#### 13.3.1 Exported Constants

None.

#### 13.3.2 Exported Access Programs

Name	In	Out	Exceptions
verify_output	$< X_{0,0}, X_{0,1}, \dots, X_{1,0}, X_{1,1}, \dots, X_{I,J} >$ where $x_{i,j} : \mathbb{R}$	$b : \mathbb{B}$	-

### 13.4 Semantics

#### 13.4.1 State Variables

None.

#### 13.4.2 Environment Variables

None.

#### 13.4.3 Assumptions

None.

#### 13.4.4 Access Routine Semantics

verify\_output():

- output: out :=
$$\begin{cases} T & \text{if transform passes verification} \\ F & \text{if transform fails verification} \end{cases}$$

To pass the verification the output matrix must pass the following conditions:

$$\Sigma_{j=0}^{freq-res} X_{i,j} \leq \Sigma x_n \quad (5)$$

where  $\Sigma x_n$  and corresponds to the portion of the signal represented by  $i$  in  $X_{i,j}$ .

#### 13.4.5 Local Functions

None.

## 14 MIS of Output Module

### 14.1 Module

output

### 14.2 Uses

Hardware Hiding Module

### 14.3 Syntax

#### 14.3.1 Exported Constants

#### 14.3.2 Exported Access Programs

Name	In	Out	Exceptions
get_output	-	$< X_{0,0}, X_{0,1}, \dots, X_{1,0}, X_{1,1}, \dots, X_{I,J} >$ where $x_{i,j} : \mathbb{R}$	-

### 14.4 Semantics

#### 14.4.1 State Variables

None.

#### 14.4.2 Environment Variables

None.

#### 14.4.3 Assumptions

1. The user requires the memory location of the output matrix, as in it does not need to be written to any external file.



#### 14.4.4 Access Routine Semantics

get\_output():

- output:  $\text{out} := \langle X_{0,0}, X_{0,1}, \dots, X_{1,0}, X_{1,1}, \dots, X_{I,J} \rangle$  where  $x_{i,j} : \mathbb{R}$   
where  $X$  is the time frequency representation of the data as calculated by Time\_Freq\_Analysis.

#### 14.4.5 Local Functions

None.

### 15 MIS of Compute Transform

#### 15.1 Module

comp\_transform

#### 15.2 Uses

Read Data Module 9, Wavelet Module 12, STFT Module 11, Zero Pad Module 16, Matrix Data Structure Module 17

#### 15.3 Syntax

##### 15.3.1 Exported Constants

None.

##### 15.3.2 Exported Access Programs

Name	In	Out	Exceptions
comp_transform	$\langle x_1, \dots, x_N \rangle$ where $x_n : \mathbb{R}, N : \mathbb{N}, f_1 : \mathbb{R}, f_2 : \mathbb{R}, T : \mathbb{R},$ $\{ 'W', 'S' \}$	$\langle X_{0,0}, X_{0,1}, \dots, X_{1,0}, X_{1,1}, \dots, X_{I,J} \rangle$ where $x_{i,j} : \mathbb{R}$	-

#### 15.4 Semantics

##### 15.4.1 State Variables

None.

##### 15.4.2 Environment Variables

None.

### 15.4.3 Assumptions

None.

### 15.4.4 Access Routine Semantics

`comp_transform()`:

- output:  $\langle X_{0,0}, X_{0,1}, \dots, X_{1,0}, X_{1,1}, \dots, X_{I,J} \rangle$  where  $x_{i,j} : \mathbb{R}$

Such that  $X$  is calculated in the following way:

1. The input signal is read using `read_data` 9 using the parameters specified by the Input Parameter Module 8.
2. The signal is zero-padded using Zero Pad Module.
3. Some computations are done regarding the boundary configuration using `calc_freq_res` and `calc_time_res` from 10 which are needed for the following step.
4. The signal is transformed using either `comp_waveletT` if  $T = 'W'$  or `comp_STFT` if  $T = 'T'$
5. The matrix output from the transforms are complex, to convert them to a real power values the matrix is  $X$  is computed from as followed:

$$X_{i,j} = \sqrt{X_C.real^2 + X_C.im^2}$$

Where  $X_C.real$  is the real part of the transform and  $X_C.im$  is the imaginary part.

### 15.4.5 Local Functions

None.

## 15.5 Considerations

The Zero-Pad Module extends the size of the signal  $x$  by 2 times the window size  $WIN\_SIZE$ , however, this doesn't affect the size of the output transform matrix.

## 16 MIS of Zero Pad Module

### 16.1 Module

`zero_pad`

### 16.2 Uses

None.

## 16.3 Syntax

### 16.3.1 Exported Constants

None.

### 16.3.2 Exported Access Programs

Name	In	Out	Exceptions
zero_pad_sig	$\langle x_1, \dots, x_N \rangle$ where $x_n : \mathbb{R}, N : \mathbb{N},$ $WIN\_SIZE$	$\langle x_1, \dots, x_{N+2WIN\_SIZE} \rangle$	-

## 16.4 Semantics

### 16.4.1 State Variables

None.

### 16.4.2 Environment Variables

None.

### 16.4.3 Assumptions

None.

### 16.4.4 Access Routine Semantics

zero\_pad\_sig():

- output:  $\langle x_1, \dots, x_{N+2WIN\_SIZE} \rangle$  such that  $x_n = 0$  from  $n[0, WIN\_SIZE]$ ,  $x_{n+WIN\_SIZE} = \hat{x}_n$  from  $n[0, N]$  where  $\hat{x}$  is the original signal and  $N$  is the length of the original signal.

### 16.4.5 Local Functions

None.

## 17 MIS of Matrix Data Structure Module

### 17.1 Module

mat

## 17.2 Uses

None.

## 17.3 Syntax

### 17.3.1 Exported Constants

None.

### 17.3.2 Exported Access Programs

Name	In	Out	Exceptions
init	$X : \mathbb{N}, Y : \mathbb{N}$	$\langle X_{0,0}, X_{0,1}, \dots, X_{1,0}, X_{1,1}, \dots, X_{X,Y} \rangle$ where $x_{x,y} : NULL$	
m	$x : \mathbb{N}, y : \mathbb{N}$	$m : \mathbb{R}$	-

## 17.4 Semantics

### 17.4.1 State Variables

None.

### 17.4.2 Environment Variables

None.

### 17.4.3 Assumptions

None.

### 17.4.4 Access Routine Semantics

mat.init():

- output:  $\langle X_{0,0}, X_{0,1}, \dots, X_{1,0}, X_{1,1}, \dots, X_{X,Y} \rangle$   
Where  $X_{x,y}$  is null but it is large able to hold type  $\mathbb{R}$ .

mat.m():

- output:  $m : \mathbb{R}$   
 $m$  is the data in the matrix at the specified index.

### 17.4.5 Local Functions

None.

## 18 MIS of Plotting Module

### 18.1 Module

plot

### 18.2 Uses

None.

### 18.3 Syntax

#### 18.3.1 Exported Constants

None.

#### 18.3.2 Exported Access Programs

Name	In	Out	Exceptions
plot_matrix	$< X_{0,0}, X_{0,1}, \dots, X_{1,0}, X_{1,1}, \dots, X_{X,Y} >$ where $x_{x,y} : \mathbb{R}$	$b : \mathbb{B}$	bad_path

### 18.4 Semantics

#### 18.4.1 State Variables

None.

#### 18.4.2 Environment Variables

None.

#### 18.4.3 Assumptions

None.

#### 18.4.4 Access Routine Semantics

plot\_matrix():

- output: out:=
$$\begin{cases} F & \text{if output file was not created sucessfully} \\ T & \text{if output file was created sucessfully} \end{cases}$$

- exception: `exc:= bad_path` if out file was not written successfully.

#### 18.4.5 Local Functions

Name	In	Out	Description
<code>calc_colour</code>	$x : \mathbb{R}$	$R, B, G : \mathbb{R}$	converts the matrix value into corresponding R,G,B values for heat map

## References

- Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of Software Engineering*. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2003.
- Daniel M. Hoffman and Paul A. Strooper. *Software Design, Automated Testing, and Maintenance: A Practical Approach*. International Thomson Computer Press, New York, NY, USA, 1995. URL <http://citeseer.ist.psu.edu/428727.html>.

## 19 Appendix