

# Module Interface Specification for SpectrumImageAnalysisPy

Isobel Bicket

November 28, 2017

# 1 Revision History

Date	Version	Notes
November 29, 2017	1.0	Initial draft

## 2 Symbols, Abbreviations and Acronyms

See **SRS** documentation at [https://github.com/icbicket/SpectrumImageAnalysisPy/blob/SpectrumImageAnalysisPy\\_dev/Doc/SRS/SRS.pdf](https://github.com/icbicket/SpectrumImageAnalysisPy/blob/SpectrumImageAnalysisPy_dev/Doc/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 Hardware Hiding Module</b>	<b>3</b>
6.1	Module . . . . .	3
6.2	Uses . . . . .	3
6.3	Syntax . . . . .	3
6.3.1	Exported Access Programs . . . . .	3
6.4	Semantics . . . . .	3
6.4.1	State Variables . . . . .	3
6.4.2	Access Routine Semantics . . . . .	3
<b>7</b>	<b>MIS of Behaviour Hiding Module</b>	<b>3</b>
7.1	Module . . . . .	3
7.2	Uses . . . . .	3
7.3	Syntax . . . . .	3
7.3.1	Exported Access Programs . . . . .	3
7.4	Semantics . . . . .	4
7.4.1	State Variables . . . . .	4
7.4.2	Access Routine Semantics . . . . .	4
<b>8</b>	<b>MIS of Import csv Module</b>	<b>4</b>
8.1	Module . . . . .	4
8.2	Uses . . . . .	4
8.3	Syntax . . . . .	4
8.3.1	Exported Access Programs . . . . .	4
8.4	Semantics . . . . .	4
8.4.1	State Variables . . . . .	4
8.4.2	Environment Variables . . . . .	4
8.4.3	Access Routine Semantics . . . . .	5
<b>9</b>	<b>MIS of Import dm3 Module</b>	<b>5</b>
9.1	Module . . . . .	5
9.2	Uses . . . . .	5
9.3	Syntax . . . . .	5

9.3.1	Exported Access Programs . . . . .	5
9.4	Semantics . . . . .	6
9.4.1	State Variables . . . . .	6
9.4.2	Environment Variables . . . . .	6
9.4.3	Access Routine Semantics . . . . .	6
<b>10</b>	<b>MIS of Import h5 Module</b>	<b>6</b>
10.1	Module . . . . .	6
10.2	Uses . . . . .	6
10.3	Syntax . . . . .	6
10.3.1	Exported Access Programs . . . . .	6
10.4	Semantics . . . . .	6
10.4.1	State Variables . . . . .	6
10.4.2	Access Routine Semantics . . . . .	6
<b>11</b>	<b>MIS of Import rpl Module</b>	<b>7</b>
11.1	Module . . . . .	7
11.2	Uses . . . . .	7
11.3	Syntax . . . . .	7
11.3.1	Exported Access Programs . . . . .	7
11.4	Semantics . . . . .	7
11.4.1	State Variables . . . . .	7
11.4.2	Access Routine Semantics . . . . .	7
<b>12</b>	<b>MIS of Export csv Module</b>	<b>7</b>
12.1	Module . . . . .	7
12.2	Uses . . . . .	7
12.3	Syntax . . . . .	7
12.3.1	Exported Access Programs . . . . .	7
12.4	Semantics . . . . .	8
12.4.1	State Variables . . . . .	8
12.4.2	Access Routine Semantics . . . . .	8
<b>13</b>	<b>MIS of Export h5 Module</b>	<b>8</b>
13.1	Module . . . . .	8
13.2	Uses . . . . .	8
13.3	Syntax . . . . .	8
13.3.1	Exported Access Programs . . . . .	8
13.4	Semantics . . . . .	9
13.4.1	State Variables . . . . .	9
13.4.2	Access Routine Semantics . . . . .	9

<b>14 MIS of Export png Module</b>	<b>9</b>
14.1 Module . . . . .	9
14.2 Uses . . . . .	9
14.3 Syntax . . . . .	9
14.3.1 Exported Access Programs . . . . .	9
14.4 Semantics . . . . .	9
14.4.1 State Variables . . . . .	9
14.4.2 Access Routine Semantics . . . . .	9
<b>15 MIS of Export rpl Module</b>	<b>10</b>
15.1 Module . . . . .	10
15.2 Uses . . . . .	10
15.3 Syntax . . . . .	10
15.3.1 Exported Access Programs . . . . .	10
15.4 Semantics . . . . .	10
15.4.1 State Variables . . . . .	10
15.4.2 Access Routine Semantics . . . . .	10
<b>16 MIS of Data Processing Richardson-Lucy Deconvolution Module</b>	<b>10</b>
16.1 Module . . . . .	10
16.2 Uses . . . . .	10
16.3 Syntax . . . . .	11
16.3.1 Exported Access Programs . . . . .	11
16.4 Semantics . . . . .	11
16.4.1 State Variables . . . . .	11
16.4.2 Access Routine Semantics . . . . .	11
<b>17 MIS of Data Processing Normalization Module</b>	<b>11</b>
17.1 Module . . . . .	11
17.2 Uses . . . . .	12
17.3 Syntax . . . . .	12
17.3.1 Exported Access Programs . . . . .	12
17.4 Semantics . . . . .	12
17.4.1 State Variables . . . . .	12
17.4.2 Access Routine Semantics . . . . .	12
<b>18 MIS of Data Processing Gain Correction Module</b>	<b>12</b>
18.1 Module . . . . .	12
18.2 Uses . . . . .	12
18.3 Syntax . . . . .	12
18.3.1 Exported Access Programs . . . . .	12
18.4 Semantics . . . . .	12
18.4.1 State Variables . . . . .	12

18.4.2	Access Routine Semantics . . . . .	12
<b>19</b>	<b>MIS of Data Processing Background Correction Module</b>	<b>13</b>
19.1	Module . . . . .	13
19.2	Uses . . . . .	13
19.3	Syntax . . . . .	13
19.3.1	Exported Access Programs . . . . .	13
19.4	Semantics . . . . .	13
19.4.1	State Variables . . . . .	13
19.4.2	Access Routine Semantics . . . . .	13
<b>20</b>	<b>MIS of Data Extraction 1D Slice Module</b>	<b>13</b>
20.1	Module . . . . .	13
20.2	Uses . . . . .	14
20.3	Syntax . . . . .	14
20.3.1	Exported Access Programs . . . . .	14
20.4	Semantics . . . . .	14
20.4.1	State Variables . . . . .	14
20.4.2	Access Routine Semantics . . . . .	14
<b>21</b>	<b>MIS of Data Extraction 2D Mask Module</b>	<b>14</b>
21.1	Module . . . . .	14
21.2	Uses . . . . .	15
21.3	Syntax . . . . .	15
21.3.1	Exported Access Programs . . . . .	15
21.4	Semantics . . . . .	15
21.4.1	State Variables . . . . .	15
21.4.2	Access Routine Semantics . . . . .	15
<b>22</b>	<b>MIS of Data Extraction 3D Mask Module</b>	<b>15</b>
22.1	Module . . . . .	15
22.2	Uses . . . . .	15
22.3	Syntax . . . . .	15
22.3.1	Exported Access Programs . . . . .	15
22.4	Semantics . . . . .	16
22.4.1	State Variables . . . . .	16
22.4.2	Access Routine Semantics . . . . .	16
<b>23</b>	<b>MIS of Display 1D Spectrum Module</b>	<b>16</b>
23.1	Module . . . . .	16
23.2	Uses . . . . .	16
23.3	Syntax . . . . .	16
23.3.1	Exported Access Programs . . . . .	16

23.4	Semantics . . . . .	16
23.4.1	State Variables . . . . .	16
23.4.2	Environment Variables . . . . .	16
23.4.3	Access Routine Semantics . . . . .	16
<b>24</b>	<b>MIS of Display 2D Image Module</b>	<b>17</b>
24.1	Module . . . . .	17
24.2	Uses . . . . .	17
24.3	Syntax . . . . .	17
24.3.1	Exported Access Programs . . . . .	17
24.4	Semantics . . . . .	17
24.4.1	State Variables . . . . .	17
24.4.2	Access Routine Semantics . . . . .	17
<b>25</b>	<b>MIS of Display 3D Spectrum Image Module</b>	<b>17</b>
25.1	Module . . . . .	17
25.2	Uses . . . . .	17
25.3	Syntax . . . . .	18
25.3.1	Exported Access Programs . . . . .	18
25.4	Semantics . . . . .	18
25.4.1	State Variables . . . . .	18
25.4.2	Environment Variables . . . . .	18
25.4.3	Access Routine Semantics . . . . .	18
<b>26</b>	<b>MIS of Data 1D Spectrum Module</b>	<b>19</b>
26.1	Template Module . . . . .	19
26.2	Uses . . . . .	19
26.3	Syntax . . . . .	19
26.3.1	Types . . . . .	19
26.3.2	Exported Access Programs . . . . .	19
26.4	Semantics . . . . .	19
26.4.1	State Variables . . . . .	19
26.4.2	Access Routine Semantics . . . . .	20
<b>27</b>	<b>MIS of Data 2D Image Module</b>	<b>20</b>
27.1	Module . . . . .	20
27.2	Uses . . . . .	20
27.3	Syntax . . . . .	21
27.3.1	Types . . . . .	21
27.3.2	Exported Access Programs . . . . .	21
27.4	Semantics . . . . .	21
27.4.1	State Variables . . . . .	21
27.4.2	Access Routine Semantics . . . . .	21



<b>28 MIS of Data 3D Spectrum Image Module</b>	<b>22</b>
28.1 Template Module . . . . .	22
28.2 Uses . . . . .	22
28.2.1 Types . . . . .	22
28.2.2 Exported Access Programs . . . . .	22
28.3 Semantics . . . . .	22
28.3.1 State Variables . . . . .	22
28.3.2 Access Routine Semantics . . . . .	23
<b>29 MIS of Array Data Structure Module</b>	<b>24</b>
29.1 Module . . . . .	24
29.2 Uses . . . . .	24
29.3 Syntax . . . . .	24
29.3.1 Exported Access Programs . . . . .	24
29.4 Semantics . . . . .	24
29.4.1 State Variables . . . . .	24
29.4.2 Access Routine Semantics . . . . .	24
<b>30 MIS of Plotting Library Module</b>	<b>25</b>
30.1 Module . . . . .	25
30.2 Uses . . . . .	25
30.3 Syntax . . . . .	25
30.3.1 Exported Access Programs . . . . .	25
30.4 Semantics . . . . .	25
30.4.1 State Variables . . . . .	25
30.4.2 Access Routine Semantics . . . . .	25
<b>31 Appendix</b>	<b>27</b>

## 3 Introduction

The following document details the Module Interface Specifications for SpectrumImageAnalysisPy, a library created for the data processing of spectrum image datasets.

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at [https://github.com/icbicket/SpectrumImageAnalysisPy/tree/SpectrumImageAnalysisPy\\_dev](https://github.com/icbicket/SpectrumImageAnalysisPy/tree/SpectrumImageAnalysisPy_dev).

## 4 Notation

[You should describe your notation. You can use what is below as a starting point. —SS]

The structure of the MIS for modules comes from [1], with the addition that template modules have been adapted from [2]. The mathematical notation comes from Chapter 3 of [1]. 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 SpectrumImageAnalysisPy.

Data Type	Notation	Description
character	char	a single symbol or digit
string	str	a sequence of characters
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 $(-\infty, \infty)$

The specification of SpectrumImageAnalysisPy 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, SpectrumImageAnalysisPy 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	Level 3
Hardware-Hiding Module		
	Import	csv dm3 h5 rpl
	Export	csv h5 png rpl
Behaviour-Hiding Module	Data processing	Richardson-Lucy Deconvolution Normalization Gain correction Background correction
	Data extraction	1D slice 2D mask 3D mask
	Display	1D spectrum plot 2D image plot 3D spectrum image plot
Software Decision Module	Data	Spectrum Image Spectrum Image
	Array Data Structure	
	Plotting Library	

Table 1: Module Hierarchy

## 6 MIS of Hardware Hiding Module

### 6.1 Module

HardwareHiding

### 6.2 Uses

### 6.3 Syntax

#### 6.3.1 Exported Access Programs

Name	In	Out	Exceptions
<a href="#">[accessProg —SS]</a>	-	-	-

### 6.4 Semantics

#### 6.4.1 State Variables

#### 6.4.2 Access Routine Semantics

[\[accessProg —SS\]](#)():

- transition: [\[if appropriate —SS\]](#)
- output: [\[if appropriate —SS\]](#)
- exception: [\[if appropriate —SS\]](#)

## 7 MIS of Behaviour Hiding Module

### 7.1 Module

BehaviourHiding

### 7.2 Uses

### 7.3 Syntax

#### 7.3.1 Exported Access Programs

Name	In	Out	Exceptions
<a href="#">[accessProg —SS]</a>	-	-	-

## 7.4 Semantics

### 7.4.1 State Variables

### 7.4.2 Access Routine Semantics

[accessProg —SS]():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

## 8 MIS of Import csv Module

### 8.1 Module

ImportCSV

### 8.2 Uses

- Data 1D Spectrum
- Array data structure
- Hardware-hiding

### 8.3 Syntax

#### 8.3.1 Exported Access Programs

Name	In	Out	Exceptions
ReadCSV	fname: str	Spectrum	NO FILE, NOT CSV

### 8.4 Semantics

#### 8.4.1 State Variables

N/A

#### 8.4.2 Environment Variables

filesystem

### 8.4.3 Access Routine Semantics

ReadCSV():

ReadCSV reads a .csv file and creates a Spectrum object with the appropriate assignments to intensity and energy range.

- input: fname: str
- transition: N/A
- output: **Spectrum**
- exceptions:

Exception	Condition
NO FILE	The filename does not correspond to any file in the filesystem $fname \notin filesystem$
NOT CSV	The indicated file is not a *.csv format $fname \notin \{files   files \in .csv\}$

## 9 MIS of Import dm3 Module

### 9.1 Module

ImportDM3

### 9.2 Uses

- Array data structure
- Hardware hiding
- Data Spectrum Image

### 9.3 Syntax

#### 9.3.1 Exported Access Programs

Name	In	Out	Exceptions
ReadDM3	filename: string	SI: $\mathbb{R}^{X \times Y \times E}$ , meta- data: dict	NO FILE, WRONG FILETYPE, NO DATA FOUND

## 9.4 Semantics

### 9.4.1 State Variables

### 9.4.2 Environment Variables

- filedm3

### 9.4.3 Access Routine Semantics

ImportDM3():

- input:
- transition:
- output:
- exception:

## 10 MIS of Import h5 Module

### 10.1 Module

ImportH5

### 10.2 Uses

### 10.3 Syntax

#### 10.3.1 Exported Access Programs

Name	In	Out	Exceptions
<a href="#">[accessProg —SS]</a>	-	-	-

## 10.4 Semantics

### 10.4.1 State Variables

### 10.4.2 Access Routine Semantics

[\[accessProg —SS\]](#)():

- transition: [\[if appropriate —SS\]](#)
- output: [\[if appropriate —SS\]](#)
- exception: [\[if appropriate —SS\]](#)

## 11 MIS of Import rpl Module

### 11.1 Module

ImportRPL

### 11.2 Uses

### 11.3 Syntax

#### 11.3.1 Exported Access Programs

Name	In	Out	Exceptions
<a href="#">[accessProg —SS]</a>	-	-	-

### 11.4 Semantics

#### 11.4.1 State Variables

#### 11.4.2 Access Routine Semantics

[\[accessProg —SS\]](#)():

- transition: [\[if appropriate —SS\]](#)
- output: [\[if appropriate —SS\]](#)
- exception: [\[if appropriate —SS\]](#)

## 12 MIS of Export csv Module

### 12.1 Module

ExportCSV

### 12.2 Uses

### 12.3 Syntax

#### 12.3.1 Exported Access Programs

Name	In	Out	Exceptions
WriteCSV	-	-	-



## 12.4 Semantics

### 12.4.1 State Variables

### 12.4.2 Access Routine Semantics

WriteCSV():

- transition: Writes data to csv file
- output: csv file
- exception:

FormatCSV():

- transition: Formats data to prepare it to write to csv file
- output: formatted data
- exception:

Verify1D():

- transition: Verifies that the input data is of the correct format (a 1D spectrum) and has a spectral range and an intensity array of equal length
- output: formatted data
- exception:

## 13 MIS of Export h5 Module

### 13.1 Module

ExportH5

### 13.2 Uses

### 13.3 Syntax

#### 13.3.1 Exported Access Programs

Name	In	Out	Exceptions
<a href="#">[accessProg—SS]</a>	-	-	-

## 13.4 Semantics

### 13.4.1 State Variables

### 13.4.2 Access Routine Semantics

[accessProg —SS]():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

## 14 MIS of Export png Module

### 14.1 Module

ExportPNG

### 14.2 Uses

### 14.3 Syntax

#### 14.3.1 Exported Access Programs

Name	In	Out	Exceptions
[accessProg —SS]	-	-	-

## 14.4 Semantics

### 14.4.1 State Variables

### 14.4.2 Access Routine Semantics

[accessProg —SS]():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

## 15 MIS of Export rpl Module

### 15.1 Module

ExportRPL

### 15.2 Uses

### 15.3 Syntax

#### 15.3.1 Exported Access Programs

Name	In	Out	Exceptions
<a href="#">[accessProg —SS]</a>	-	-	-

### 15.4 Semantics

#### 15.4.1 State Variables

#### 15.4.2 Access Routine Semantics

[\[accessProg —SS\]](#)():

- transition: [\[if appropriate —SS\]](#)
- output: [\[if appropriate —SS\]](#)
- exception: [\[if appropriate —SS\]](#)

## 16 MIS of Data Processing Richardson-Lucy Deconvolution Module

### 16.1 Module

RLDeconvolution

### 16.2 Uses

Array Data Structure

## 16.3 Syntax

### 16.3.1 Exported Access Programs

Name	In	Out	Exceptions
RLDeconvolution	S, iterations, threads	S, deconvolved S	-
SIDeconvolution	-	-	-

## 16.4 Semantics

### 16.4.1 State Variables

N/A

### 16.4.2 Access Routine Semantics

RLDeconvolution():

- input: S, S, iterations, threads
- transition:
- output: deconvolved spectrum
- exception: Divide by zero!

SIDeconvolution():

- input: SI, iterations, S, threads
- transition:
- output: Deconvolved spectrum image
- exception: divide by zero

## 17 MIS of Data Processing Normalization Module

### 17.1 Module

Normalization

## 17.2 Uses

## 17.3 Syntax

### 17.3.1 Exported Access Programs

Name	In	Out	Exceptions
<a href="#">[accessProg —SS]</a>	-	-	-

## 17.4 Semantics

### 17.4.1 State Variables

### 17.4.2 Access Routine Semantics

[\[accessProg —SS\]](#)():

- transition: [\[if appropriate —SS\]](#)
- output: [\[if appropriate —SS\]](#)
- exception: [\[if appropriate —SS\]](#)

# 18 MIS of Data Processing Gain Correction Module

## 18.1 Module

GainCorr

## 18.2 Uses

## 18.3 Syntax

### 18.3.1 Exported Access Programs

Name	In	Out	Exceptions
<a href="#">[accessProg —SS]</a>	-	-	-

## 18.4 Semantics

### 18.4.1 State Variables

### 18.4.2 Access Routine Semantics

[\[accessProg —SS\]](#)():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

## 19 MIS of Data Processing Background Correction Module

### 19.1 Module

BackgroundCorr

### 19.2 Uses

### 19.3 Syntax

#### 19.3.1 Exported Access Programs

Name	In	Out	Exceptions
[accessProg —SS]	-	-	-

### 19.4 Semantics

#### 19.4.1 State Variables

#### 19.4.2 Access Routine Semantics

[accessProg —SS]():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

## 20 MIS of Data Extraction 1D Slice Module

### 20.1 Module

Slice1D

## 20.2 Uses

## 20.3 Syntax

### 20.3.1 Exported Access Programs

Name	In	Out	Exceptions
CreateMask	-	-	-
ApplyMask	-	-	-

## 20.4 Semantics

### 20.4.1 State Variables

- Mask (2D array of booleans)

### 20.4.2 Access Routine Semantics

CreateMask():

- transition: Creation of the mask for a 2d dataset - relies on user interaction
- output:
- exception:

[should this be here, or in display? —Author]

ApplyMask():

- transition: Applies 2d mask to dataset
- output:
- exception:

# 21 MIS of Data Extraction 2D Mask Module

## 21.1 Module

Mask2D

## 21.2 Uses

## 21.3 Syntax

### 21.3.1 Exported Access Programs

Name	In	Out	Exceptions
Create mask	keyboard event, mouse event, data size	2d bool mask of data size	-
Apply mask			
Modify mask			

## 21.4 Semantics

### 21.4.1 State Variables

- mask2D

### 21.4.2 Access Routine Semantics

[accessProg —SS]():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

# 22 MIS of Data Extraction 3D Mask Module

## 22.1 Module

Mask3D

## 22.2 Uses

## 22.3 Syntax

### 22.3.1 Exported Access Programs

Name	In	Out	Exceptions
[accessProg —SS]	-	-	-



## 22.4 Semantics

### 22.4.1 State Variables

mask3d

### 22.4.2 Access Routine Semantics

[accessProg —SS]():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

## 23 MIS of Display 1D Spectrum Module

### 23.1 Module

Disp1D

### 23.2 Uses

Data 1D Spectrum Plotting library

### 23.3 Syntax

#### 23.3.1 Exported Access Programs

Name	In	Out	Exceptions
plot	-	-	-

## 23.4 Semantics

### 23.4.1 State Variables

### 23.4.2 Environment Variables

fig

### 23.4.3 Access Routine Semantics

[accessProg —SS]():

- transition: [if appropriate —SS]

- output: [if appropriate —SS]
- exception: [if appropriate —SS]

## 24 MIS of Display 2D Image Module

### 24.1 Module

Disp2D

### 24.2 Uses

### 24.3 Syntax

#### 24.3.1 Exported Access Programs

Name	In	Out	Exceptions
[accessProg —SS]	-	-	-

### 24.4 Semantics

#### 24.4.1 State Variables

#### 24.4.2 Access Routine Semantics

[accessProg —SS]():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

## 25 MIS of Display 3D Spectrum Image Module

### 25.1 Module

Disp3D

### 25.2 Uses

- Data
- Plotting library

- 2D image plot
- 1D spectrum plot

## 25.3 Syntax

### 25.3.1 Exported Access Programs

Name	In	Out	Exceptions
<a href="#">[accessProg —SS]</a>	-	-	-

## 25.4 Semantics

### 25.4.1 State Variables

- axis2D image
- axis1D spectrum
- axis2D mask
- axis1D contrast
- axis colourbar
- polygons
- slicer

[do polygons and slicer belong here, or in the mask2d and slice1d modules? —Author]

### 25.4.2 Environment Variables

- Plotting window displayed on screen
- Keyboard keys and mouse buttons

### 25.4.3 Access Routine Semantics

[\[accessProg —SS\]](#)():

- transition: [\[if appropriate —SS\]](#)
- output: [\[if appropriate —SS\]](#)
- exception: [\[if appropriate —SS\]](#)

,

## 26 MIS of Data 1D Spectrum Module

### 26.1 Template Module

Spectrum

### 26.2 Uses

- Array data structure

### 26.3 Syntax

#### 26.3.1 Types

Spectrum

#### 26.3.2 Exported Access Programs

Name	In	Out	Exceptions
init	data, (Srange   (dis- person & [index, value])), Slabel, Sunit	-	WRONG DATA TYPE, LENGTH MISMATCH

### 26.4 Semantics

This module contains the abstract data type Spectrum, including the following state variables.

#### 26.4.1 State Variables

- $SRange$ :  $\mathbb{R}^K$
- $data$ :  $\mathbb{R}^K$
- $index$ :  $\mathbb{Z}$
- $value$ :  $\mathbb{R}$
- $dispersion$ :  $\mathbb{R}$
- $Slabel$ :  $str$
- $Sunit$ :  $str$
- $metadata$ :  $dict$

### 26.4.2 Access Routine Semantics

`init()`: `init` initializes a `Spectrum` object.

- input:
  - *data*: intensity values,  $\in \mathbb{R}^K$
  - *Srange*: spectral axis values,  $\in \mathbb{R}^K$
  - *dispersion*: difference between neighbouring channels along the spectral axis,  $\mathbb{R}$
  - *index*: location on the spectral axis at which *value* is,  $\mathbb{Z}$
  - *value*: value of the spectral axis (in spectral axis units) at the location given by *index*,  $\mathbb{R}$
  - *Slabel*: spectrum label, the name for the spectral axis (*e.g.* Energy, Wavelength), *str*
  - *Sunit*: spectrum units, the units which the spectral axis uses (*e.g.* eV, nm), *str*
- transition: Creates all state variables
- output: N/A
- exception:

Exception	Condition
WRONG DATA TYPE	Any of the input data are the wrong type
LENGTH MIS-MATCH	The length of <i>Srange</i> is not the same as the length of <i>data</i> $len(Srange) \neq len(data)$

## 27 MIS of Data 2D Image Module

### 27.1 Module

Image

### 27.2 Uses

- Array data structure

## 27.3 Syntax

### 27.3.1 Types

Image

### 27.3.2 Exported Access Programs

Name	In	Out	Exceptions
init	data, Imcal, metadata	-	WRONG DATA TYPE, WRONG DIMENSIONS

## 27.4 Semantics

This module contains the abstract data type Spectrum, including the following state variables.

### 27.4.1 State Variables

- data:  $\mathbb{R}^{X \times Y}$
- Imcal:  $\mathbb{R}$
- metadata: dict

### 27.4.2 Access Routine Semantics

init(): init initializes an Image object.

- input:
  - *data*: intensity values,  $\in \mathbb{R}^{X \times Y}$
  - *Imcal*: image calibration values (*e.g.* number of nm per pixel),  $\in \mathbb{R}$
  - *metadata*: dictionary containing extra information about the source of the image (*e.g.* experimental parameters)
- transition: Creates all state variables
- output: N/A
- exception:

Exception		Condition
WRONG TYPE	DATA	The input data are not real numbers or the Imcal value is not a real float $data \notin \mathbb{R}^{X \times Y}   Imcal \notin \mathbb{R} \Rightarrow \text{WRONG\_DATA\_TYPE}$
WRONG SIONS	DIMEN-	The input data is not 2D $size(data) \notin \mathbb{N}^2 \Rightarrow \text{WRONG\_DIMENSIONS}$

## 28 MIS of Data 3D Spectrum Image Module

### 28.1 Template Module

SI

### 28.2 Uses

- Array Data Structure

#### 28.2.1 Types

Spectrum Image

#### 28.2.2 Exported Access Programs

Name	In	Out	Exceptions
init	data, Srange   disper- sion & [index, value], Slabel, Sunit, Imcal, metadata	-	WRONG DATA TYPE, WRONG DIMENSIONS

### 28.3 Semantics

This module holds spectrum image data (a 3D dataset) and associated calibrations and other related information.

#### 28.3.1 State Variables

- $data: \mathbb{R}^{X \times Y \times K}$
- $Imcal: \mathbb{R}$

- *dispersion*:  $\mathbb{R}$
- *Srange*:  $\mathbb{R}^K$
- *index*:  $\mathbb{Z}$
- *value*:  $\mathbb{R}$
- *Slabel*: string
- *Sunit*: string
- *metadata*: dict

### 28.3.2 Access Routine Semantics

init

- input:
  - *data*: intensity values,  $\in \mathbb{R}^{X \times Y \times K}$
  - *Srange*: spectral axis values,  $\in \mathbb{R}^K$
  - *dispersion*: difference between neighbouring channels along the spectral axis,  $\mathbb{R}$
  - *index*: location on the spectral axis at which *value* is,  $\mathbb{Z}$
  - *value*: value of the spectral axis (in spectral axis units) at the location given by *index*,  $\mathbb{R}$
  - *Slabel*: spectrum label, the name for the spectral axis (*e.g.* Energy, Wavelength), *str*
  - *Sunit*: spectrum units, the units which the spectral axis uses (*e.g.* eV, nm), *str*
  - *Imcal*: image calibration values (*e.g.* number of nm per pixel),  $\in \mathbb{R}$
  - *metadata*: dictionary containing extra information about the source of the image (*e.g.* experimental parameters)
- transition: Initialize all state variables
- output: N/A
- exception:



Exception	Condition
WRONG DATA TYPE	Any of the input data are the wrong type $(data \notin \mathbb{R}^{X \times Y \times K}) \vee (Srange \notin \mathbb{R}^K) \vee (dispersion \notin \mathbb{R}) \vee (index \notin \mathbb{Z}) \vee (value \notin \mathbb{R}) \vee (Slabel \notin str) \vee (Sunit \notin str) \vee (Imcal \notin \mathbb{R}) \Rightarrow \text{WRONG\_DATA\_TYPE}$
LENGTH MIS-MATCH	The length of Srange is not the same as the length of data's spectral axis $len(Srange) \neq size(data)[2] \Rightarrow \text{LENGTH\_MISMATCH}$
WRONG DATA TYPE	The input data are not real numbers or the Imcal value is not a real float $data \notin \mathbb{R}^{X \times Y} \vee Imcal \notin \mathbb{R} \Rightarrow \text{WRONG\_DATA\_TYPE}$
WRONG DIMENSIONS	The input data is not 2D $size(data) \notin \mathbb{N}^2 \Rightarrow \text{WRONG\_DIMENSIONS}$

## 29 MIS of Array Data Structure Module

### 29.1 Module

Array

### 29.2 Uses

### 29.3 Syntax

#### 29.3.1 Exported Access Programs

Name	In	Out	Exceptions
<a href="#">[accessProg —SS]</a>	-	-	-

### 29.4 Semantics

#### 29.4.1 State Variables

#### 29.4.2 Access Routine Semantics

[\[accessProg —SS\]](#)():

- transition: [\[if appropriate —SS\]](#)

- output: [if appropriate —SS]
- exception: [if appropriate —SS]

## 30 MIS of Plotting Library Module

### 30.1 Module

Plotting

### 30.2 Uses

### 30.3 Syntax

#### 30.3.1 Exported Access Programs

Name	In	Out	Exceptions
[accessProg —SS]	-	-	-

### 30.4 Semantics

#### 30.4.1 State Variables

#### 30.4.2 Access Routine Semantics

[accessProg —SS]():

- transition: [if appropriate —SS]
- output: [if appropriate —SS]
- exception: [if appropriate —SS]

## References

- [1] D. M. Hoffman and P. A. Strooper, *Software Design, Automated Testing, and Maintenance: A Practical Approach*. New York, NY, USA: International Thomson Computer Press, 1995.
- [2] C. Ghezzi, M. Jazayeri, and D. Mandrioli, *Fundamentals of Software Engineering*. Upper Saddle River, NJ, USA: Prentice Hall, 2nd ed., 2003.

## 31 Appendix

[Extra information if required —SS]