

|  |
| --- |
|  |
| MathEO  Requirements  Documentation  Version 1.0 Draft1 | |
| Kavya Jagan, Pieter De Vis and Sam Hunt |
| 28/1/2021 |
|  |

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Description | Author |
| 1.0 | 04/02/2021 | Initial requirements document | Kavya Jagan, Pieter De Vis and Sam Hunt |
|  |  |  |  |

Version History

# Introduction

## Terminology

Terminology used in this document.

## References

References to external documents as well as the statement of work, design document, …

# User Requirements

Software should provide access to mathematic algorithms that may be of use in Earth Observation data processing and analysis.

# Functional Requirements

Requirements are graded as follows:

* *Critical:* Core to the software, must be met.
* *Major:* Improves the software, should be met.
* *Minor:* Useful, but not critical or major. If cannot be implemented in a first release perhaps can be implemented later.

## General requirements

### [Critical] The software must contain python implementations of mathematical algorithms

### [Minor] Algorithms should be implemented generically, to apply to the widest range of possible problems.

## User interface

Definition of those user interface characteristics that allow to understand and learn the software easily so the user be able to perform his/her tasks efficiently including the interface exemplar description.

### [Major] The software should be an importable, packaged Python module

## External interface

Definition of interfaces with other software or hardware.

### [Critical] The software should be importable within other group software.

### [Critical] Interfaces to functions/class methods (input arguments, returns values) should be well defined and document.

### [Critical] Input and output data should be numpy arrays where possible.

## Input / Output File(s)

The contents of the files that the software will read in/save results to.

### [Minor] Any necessary output data should be in well-defined format, preferably netcdf4.

## Mathematical

Initial list of algorithms the software should implement:

### [Critical] Tensor product algebra

### [Critical] Toeplitz matrix multiplication using FFT

### [Critical] Conversion between covariance and correlation matrices and uncertainties

### [Critical] Cholesky decomposition (nearest positive, definite)

### [Critical] Evaluating Jacobian matrices

### [Critical] Multiplication of a Kronecker product of matrices with a matrix

## Operational\*

Hardware, operating system, memory requirements, performance, efficiency, portability etc.

### [Major] Algorithms should be designed and tested to minimise run time

### [Major] Algorithms should be designed to not use more RAM than necessary. Use on a standard NPL laptop should be targeted where feasible and sufficient for testing.

### [Minor] Software should be easily installable through pip

## Reliability\*

Specification of the software execution level concerning the maturity, fault tolerance and recovery.

### [Minor] Reach at least 95% MC/DC coverage.

## Maintenance\*

Description of the elements facilitating the understanding and execution of the future *Software* modifications.

### [Minor] Software should be regular updated using continuous integration methods

## Design and construction limitations/constraints\*

Needs, timelines imposed by the Customer.

### [Minor] Where possible the software should be implemented in an Object Oriented manner

## Legal and regulative\*

Needs imposed by laws, regulations, NPL security or IP regulations. To be defined following NPL policy decision on software availability.

### [Critical]

# Appendix I: Algorithm description\*