|  |
| --- |
|  |

**MUCP Algorithm MANUAL**

**by**

**Kirodh Boodhraj**

**Handed over to DFFE on 2 October 2025**

|  |  |
| --- | --- |
| Executive Summary | |
| Title: | MUCP Algorithm Manual (Kirodh Boodhraj) |
| Abstract: | MUCP Algorithms. |
| Keywords: |  |
| Objective: | Manual written for DFFE and future users. |
| Purpose: |  |
| Methodology: |  |
| Conclusions: |  |
| Limitations or Risks: |  |
| Benefits: |  |
| Contract Context: | This document is the result of a research effort funded by NRM and CSIR PG internal funding. |

|  |  |
| --- | --- |
| Approval Page | |
| Project Name | MUCP |
| Project No |  |
| Title |  |
| Prepared for |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Authored for | CSIR | Signature | Date |
| By | Kirodh Boodhraj | Kirodh | 30 Sept. 25 |
| Designation |  | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Change History | | | | |
| Change Control No | Date | Name | Description |
| Version 1 | 27 Jun. 23 |  | Original release. |
| Version 2 | 30 Sept. 25 |  | Cleaned up the document. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Contents

[Abbreviations 5](#_Toc210289347)

[1 Introduction 6](#_Toc210289348)

[2 User support data processing 7](#_Toc210289349)

[2.1 User Data Pre-Processing Manual 7](#_Toc210289350)

[2.2 GIS Mapping Shapefile 7](#_Toc210289351)

[2.3 NBAL Shapefile 8](#_Toc210289352)

[2.4 MIU Shapefile 9](#_Toc210289353)

[2.5 Compartment Shapefile 9](#_Toc210289354)

[2.6 MIU Linked Species Excel File 10](#_Toc210289355)

[2.7 NBAL Linked Species Excel File 10](#_Toc210289356)

[2.8 Compartment Prioritization CSV File 11](#_Toc210289357)

[2.9 Summary of Pre-Processing Logic 11](#_Toc210289358)

[3 Prioritization 12](#_Toc210289359)

[3.1 Theoretical Background 12](#_Toc210289360)

[3.2 Key Properties 12](#_Toc210289361)

[3.3 Implementation Steps 13](#_Toc210289362)

[3.4 Worked Example 15](#_Toc210289363)

[3.5 Notes and Recommendations 16](#_Toc210289364)

[4 Density 17](#_Toc210289365)

[4.1 Theoretical Background 17](#_Toc210289366)

[4.2 Variables Used in Density Calculation 17](#_Toc210289367)

[4.3 Algorithm Steps 18](#_Toc210289368)

[4.4 Worked Example 19](#_Toc210289369)

[4.5 Notes and Recommendations 20](#_Toc210289370)

[5 Person Days 21](#_Toc210289371)

[5.1 Person Days Algorithm 21](#_Toc210289372)

[5.2 Normal Person Days (Base Calculation) 21](#_Toc210289373)

[5.2.1 Theory 21](#_Toc210289374)

[5.3 Adjusted Person Days 23](#_Toc210289375)

[5.3.1 Theory 23](#_Toc210289376)

[5.4 Example Calculation 25](#_Toc210289377)

[5.5 Notes and Recommendations 26](#_Toc210289378)

[6 Costing 27](#_Toc210289379)

[6.1 Parameters and Inputs 27](#_Toc210289380)

[6.1.1 Cost Model Parameters 27](#_Toc210289381)

[6.1.2 Supporting Data 27](#_Toc210289382)

[6.2 Algorithm Workflow 28](#_Toc210289383)

[6.2.1 Annual Iteration 28](#_Toc210289384)

[6.3 Preprocessing 28](#_Toc210289385)

[6.4 For year 1 28](#_Toc210289386)

[6.4.1 Initial entries calculations 28](#_Toc210289387)

[6.4.2 Follow-up entries calculations 30](#_Toc210289388)

[6.5 For year 2 onwards 31](#_Toc210289389)

[6.6 Budget Allocation after costing was calculated per compartment 33](#_Toc210289390)

[6.6.1 Optimal Budget 33](#_Toc210289391)

[6.6.2 Constrained Budgets (Budgets 1–4) 33](#_Toc210289392)

[6.7 Budget Escalation 33](#_Toc210289393)

[6.8 Assumptions 34](#_Toc210289394)

[6.9 Example 34](#_Toc210289395)

[6.9.1 Year 1 34](#_Toc210289396)

[6.9.2 Year 2 onwards 35](#_Toc210289397)

[6.10 Notes and Recommendations 36](#_Toc210289398)

[7 Flow 37](#_Toc210289399)

[7.1 Input Data Requirements 37](#_Toc210289400)

[7.2 Algorithm Theory 37](#_Toc210289401)

[7.3 Algorithm Description 38](#_Toc210289402)

[7.3.1 ### Step A: Determine Flow Factor by Age Class 38](#_Toc210289403)

[7.3.2 ### Step B: Obtain the MAR per compartment, area and density 38](#_Toc210289404)

[7.3.3 ### Step C: Year 1 calculation 39](#_Toc210289405)

[7.3.4 ### Step D: Year 2 and further calculation 40](#_Toc210289406)

[7.4 Worked Example 40](#_Toc210289407)

[7.5 Assumptions 41](#_Toc210289408)

[7.6 Notes and Recommendations 41](#_Toc210289409)

[8 Main costing loop 42](#_Toc210289410)

[8.1 Initial Time Step (Year 1) 42](#_Toc210289411)

[8.2 Subsequent Time Steps (Year 2 onwards) 43](#_Toc210289412)

[8.3 Budget Optimization and Masking 43](#_Toc210289413)

[8.4 Data Preparation 43](#_Toc210289414)

[8.5 Calculation Sequence Within Each Time Step 44](#_Toc210289415)

[8.6 Example Input Data Structures 45](#_Toc210289416)

[8.6.1 GIS Mapping, NBAL, MIU and compartment shapefiles 45](#_Toc210289417)

[8.6.2 MIU and NBAL Linked Species Excel files 46](#_Toc210289418)

[8.6.3 Compartment Priorities CSV file 47](#_Toc210289419)

[8.6.4 Species Attributes (support data) 48](#_Toc210289420)

[8.6.5 Costing Data (support data) 49](#_Toc210289421)

[9 Release notes & references 50](#_Toc210289422)

[9.1 Release notes 50](#_Toc210289423)

[9.1.1 Algorithmic Differences 50](#_Toc210289424)

[9.1.2 Algorithm Derivation Methodology 51](#_Toc210289425)

[9.1.3 Functional and Structural Differences 51](#_Toc210289426)

[9.1.4 Technical Implementation 52](#_Toc210289427)

[9.1.5 Beta Release 52](#_Toc210289428)

[9.2 References 52](#_Toc210289429)

Abbreviations

|  |  |
| --- | --- |
| CSIR | Council for Scientific and Industrial Research |
| DFFE | Department of Forestry, Fisheries and Environment |
| MUCP | Management Unit Control Plan |
|  |  |
|  |  |
|  |  |
|  |  |

# Introduction

Le Maitre et al. (2012) developed a generic species and area prioritization model (MUCP Tool) for use in prioritising invasive alien plant control operations in South Africa using spatial data and Forsyth et al. (2012) . It schedules treatments of invasions in the catchment taking into account the current state of the invasions, benefits of clearing, treatments that are required and the resources provided in its budget. The MUCP tool does not generate a detailed annual schedule of annual operations but the DEA-Natural Resources Management (NRM) programme has an Annual Plan of Operations tool which serves that purpose.

This manual will guide the user on the algorithms constituting the MUCP Tool as well as the users input support data.

There are **5 main algorithms that constitute the MUCP code engine**:

* **Prioritization (arbritrary)**
* **Density (%)**
* **Person Days [normal and adjusted] (arbritrary)**
* **Costing (Currency)**
* **Flow (M^3/s)**

Before the algorithms can be run, the users’ support data needs to be pre-processed.

The algorithms were coded in Python (version 3.11 latest at the time of this manual), using vectorized array calculations with Pandas, Numpy, Regex, and Geopandas in order to speed up the algorithm run times.

A full test case with approximately 400 rows for a 50 year plan runs in just under 10 seconds as a benchmark.

Please note that these algorithms were derived from sensitivity analyses of the desktop version of the tool, where parameter values were systematically varied and equations reconstructed through brute-force testing. As such, while the implementation reflects the best possible alignment with the original system, slight deviations may occur due to reconstruction uncertainties.

# User support data processing

## User Data Pre-Processing Manual

This section describes the data pre-processing requirements and validation rules for all user-supplied input files. The pre-processing step ensures that the data is consistent, conforms to the required standards, and is ready for subsequent algorithmic processing in the tool.

There are **seven user input files** required:

1. GIS Mapping Shapefile

2. NBAL Shapefile

3. MIU Shapefile

4. Compartment Shapefile

5. MIU Linked Species Excel File

6. NBAL Linked Species Excel File

7. Compartment Prioritization CSV File

General Rules:

\* All **area values** in all files must be strictly positive.

\* Where duplicate identifiers occur, the first valid entry is retained, and subsequent duplicates are ignored with a warning to the user. The exception is in the gis mapping shapefile where the areas are added if there are any duplicate entries.

## GIS Mapping Shapefile

**Expected Columns:**

\* `NBAL\_ID`

\* `MIU\_ID`

\* `COMPT\_ID`

\* `AREA`

**Pre-processing Rules:**

1. If `NBAL\_ID`, `MIU\_ID`, and `COMPT\_ID` have the same identifiers, coalesce them into a single entry by summing their corresponding `AREA` values.

2. If an `NBAL\_ID` is present, a corresponding `MIU\_ID` must also exist. If missing, issue a **warning** to the user.

3. If an `NBAL\_ID` is present but no corresponding `MIU\_ID` exists within the associated `COMPT\_ID`, issue a **warning** to the user as this entry cannot be used, it must be discarded.

## NBAL Shapefile

**Expected Columns:**

\* `NBAL\_ID`

\* `STAGE`

\* `FIRST\_DATE`

\* `LAST\_DATE`

\* `AREA`

\* `CONTRACTID`

**Pre-processing Rules:**

1. Each `NBAL\_ID` must be unique. If duplicates exist, retain only the first occurrence.

2. The `STAGE` column must be normalized as follows:

\* `"Initial Treatment"` -> `0`

\* `"Maintenance"` -> `-1`

\* `"Nth Follow Up"` (e.g. `"5th Follow Up"`) -> `1` (for any numeric follow-up)

3. The `AREA` column must be cast to a float and validated to be positive.

4. Columns other than the required ones are ignored during pre-processing.

## MIU Shapefile

**Expected Columns:**

\* `MIU\_ID`

\* `AREA`

\* `RIPARIAN\_C`

**Pre-processing Rules:**

1. Each `MIU\_ID` must be unique. If duplicates exist, retain only the first occurrence.

2. `RIPARIAN\_C` must be standardized to one of:

\* "l" -> dryland

\* "r" -> riparian

3. `AREA` must be cast to a float and validated to be positive.

## Compartment Shapefile

**Expected Columns:**

\* `COMPT\_ID`

\* `AREA\_HA`

\* `SLOPE`

\* `WALK\_TIME`

\* `DRIVE\_TIME`

\* `TERRAIN`

\* `COSTING`

\* `GROW\_CON`

**Pre-processing Rules:**

1. Each `COMPT\_ID` must be unique. If duplicates exist, retain only the first occurrence.

2. `AREA\_HA` must be cast to a float and validated to be positive.

3. `SLOPE` must be between **0 and 90 degrees**, inclusive. Negative or out-of-range values are invalid.

4. `WALK\_TIME` and `DRIVE\_TIME` must be positive values (measured in minutes).

5. `COSTING` must be a positive integer.

6. `GROW\_CON` must exactly match one of the pre-defined growth conditions in the **support reference data**. If no match is found, the entry is rejected.

## MIU Linked Species Excel File

**Expected Columns:**

\* `MIU\_ID`

\* `Species`

\* `Idenscode`

\* `Age`

**Pre-processing Rules:**

1. `MIU\_ID` values must be present in the GIS Mapping Shapefile’s `MIU\_ID` column. If not found, issue a **warning** to the user.

2. `Idenscode` must be a float or integer between **0 and 100**, inclusive. Values outside this range are invalid.

## NBAL Linked Species Excel File

**Expected Columns:**

\* `NBAL\_ID`

\* `Species`

\* `Idenscode`

\* `Age`

**Pre-processing Rules:**

1. `NBAL\_ID` values must be present in the GIS Mapping Shapefile’s `NBAL\_ID` column. If not found, issue a **warning** to the user.

2. `Idenscode` must be a float or integer between **0 and 100**, inclusive. Values outside this range are invalid.

## Compartment Prioritization CSV File

**Expected Columns:**

\* Mandatory: `COMPT\_ID`

\* Additional: All other column headers must correspond to variables defined in the **support prioritization data** of the tool.

**Pre-processing Rules:**

1. `COMPT\_ID` must be unique. If duplicates exist, retain only the first occurrence.

2. Each `COMPT\_ID` must exist in the GIS Mapping Shapefile `COMPT\_ID` column. If missing, issue a **warning** to the user.

3. All additional columns must match valid prioritization variables from the support data; otherwise, reject with an error.

## **Summary of Pre-Processing Logic**

\* Duplicate identifiers are reduced by retaining the first valid entry, with exception to the gis mapping shapefile entries.

\* All numeric values (area, slope, times, idenscode) must be positive and within their expected ranges.

\* Referential integrity must be maintained between shapefiles (e.g. `NBAL\_ID` in GIS Mapping must match `NBAL\_ID` in NBAL Linked Species).

\* Invalid or missing references trigger warnings or errors depending on severity.

# Prioritization

## Theoretical Background

The **prioritization algorithm** is designed to combine multiple environmental and management variables into a single **compartment prioritization score**. This score determines the relative importance of a compartment for treatment, intervention, or monitoring.

The algorithm is based on **multi-criteria decision analysis (MCDA)** principles. Each variable (e.g., **elevation**, **aggression**, **slope**, etc.) is:

1. Mapped to a priority category (based on pre-defined ranges in the reference table).

2. Weighted according to user preferences (via the graphical user interface).

3. Aggregated across all selected variables to produce a final prioritization score.

## Key Properties

\* **Weights do not need to sum to 1.**

\* In practice, weights may sum to a value greater than 1 or less than 1.

\* However, for more stable results, the sum should preferably be **greater than 1**.

\* If the sum of weights is less than 1, a **user warning** should be generated to avoid under-representing priorities.

\* **Prioritization Categories** are discrete ranges that map raw data values (e.g., **Elevation = 3**) to integer priority levels (e.g., **Priority = 1**).

\* **Final Prioritization Score** is computed as the **weighted sum of all category priorities**.

**Prioritization\_Compartment** = sum of **P\_category x W\_category** for all categories per that compartment

Where:

\* **P\_category** = Priority category value assigned to current category in compartment based on the value form the csv priorities user file.

\* **W\_category** = User-defined weighting for category.

## Implementation Steps

The following procedure is applied to compute the prioritization score for each compartment:

1. Iterate Through Compartments

\* Read each compartment ID from the **compartment prioritization CSV file**.

\* Example: Compartment ID `H\_123`.

2. Loop Through Variable Columns

\* For each column (header) in the CSV file (e.g., `Elevation`, `Aggression`), process the variable individually.

3. Match Column Header to Reference Table

\* Cross-reference the column header against the Support Priority Fields Table.

\* Extract the internal representation of the variable name used by the tool.

\* Example: `"Elevation"` -> `"elevation"`, `"Aggression"` -> `"aggressive"`. Note that no more mapping is needed for the web tool, the user just needs to match the priority category with the category they create.

4. Retrieve User Weighting Values

\* Match the variable name to the **user-selected weighting values** in the GUI input file.

\* Example:

\* Elevation weighting = `0.9`

\* Aggression weighting = `0.2`

5. Assign Priority Category

\* For each variable, determine the **priority category** based on its raw value and the range definitions in the reference table.

\* Example category ranges:

|  |  |  |  |
| --- | --- | --- | --- |
| elevation | | Aggressive | |
| Range | priority | range | priority |
| 0-4 | 1 | 0-30 | 1 |
| 5-10 | 2 | 31-60 | 2 |

\* If Elevation = 3 -> Priority = 1.

\* If Aggression = 50 -> Priority = 2.

6. Multiply by Weighting

\* Multiply each category priority by the associated user weighting.

\* Example:

\* Elevation: 1 x 0.9 = 0.9

\* Aggression: 2 x 0.2 = 0.4

7. Aggregate Priorities

\* Sum across all weighted priorities for the compartment.

\* Example:

Final\_prioritization = 0.9 + 0.4 = 1.3

8. Assign Compartment Prioritization Score

\* Record the calculated score for compartment `H\_123`.

## Worked Example

**Input Data:**

|  |  |  |
| --- | --- | --- |
| Compartment | Elevation | Aggression |
| H\_123 | 3 | 50 |

**Reference Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| elevation | | Aggressive | |
| Range | priority | range | priority |
| 0-4 | 1 | 0-30 | 1 |
| 5-10 | 2 | 31-60 | 2 |

**User-Defined Weights:**

|  |  |
| --- | --- |
| Variable | Weight |
| Elevation | 0.9 |
| Aggression | 0.2 |

**Step-by-Step Calculation:**

1. Elevation = 3 -> falls in range 0 - 4 -> Priority = 1.

\* Weighted Score = 1 x 0.9 = 0.9.

2. Aggression = 50 -> falls in range 31 - 60 -> Priority = 2.

\* Weighted Score = 2 x 0.2 = 0.4.

3. Aggregate:

Final Prioritization Score for H = 0.9 + 0.4 = 1.3

## Notes and Recommendations

\* **Weighting Guidance**: While weights may sum to less than or greater than 1, a sum greater than 1 is recommended for balanced prioritization. Implement a **user warning** if weights sum to less than 1.

\* **Flexibility**: The algorithm supports any number of prioritization variables, provided corresponding category ranges are defined. It is limited to maximum 6 for the viewer and for practicality.

\* **Reproducibility**: Since the algorithm is deterministic, repeated runs with identical inputs will always yield the same prioritization scores.

\* **Transparency**: All input data (compartment values, reference tables, and weights) are stored in CSV/GUI input files, ensuring full traceability of results.

# Density

## Theoretical Background

The **density algorithm** calculates the effective stand density of invasive alien species within compartments, incorporating treatment history, species-specific reduction factors, and temporal propagation across budget years.

The algorithm is designed to:

\* Capture the interaction between MIU-linked species records and NBAL-linked species records, where applicable.

\* Apply initial and follow-up reduction factors depending on whether it is the first or subsequent treatment year.

\* Adjust densities based on species mapped attributes i.e. Idenscode or density.

\* Incorporate treatment frequency, allowing for multiple treatments per year or less frequent treatments across multi-year intervals.

\* Propagate densities across a xyz number of years budgeting horizon, including densification or density reduction effects where density increases or the compartment is cleared respectively between treatments.

The overall principle is that density in each year is derived from previous-year densities, reduced (or sometimes modified) by follow-up factors, with adjustments for MIU and NBAL records per species level.

## Variables Used in Density Calculation

The algorithm draws on the following variables:

\* Treatment frequency (from compartment metadata).

\* Idenscode (species density index) from both NBAL and MIU linked species file, used for the initial start of simulation i.e. timestep 0.

\* Initial reduction density factor (from species tab).

\* Follow-up reduction density factor (from species tab).

\* Densification factor (from species tab, applied as a post-processing step each year to compartments not cleared).

The density can be densify or reduce mode depending on the sign of the density factor used:

Densify = density\_previous x (100 + f) /100

reduceDensity = density\_previous x (100 – f) / 100

, where f is the density reduction or densify factor.

## Algorithm Steps

Assuming that all the densify and reduction for initial and follow-up are a positive integer. We need to then handle the following cases.

### Case 1: Initial reduction cases for initial timestep in main costing loop

This case includes entries in the gis mapping file that have only the **compartment and miu id**. It also includes the **compartment, miu and nbal id** entries but only if the treatment is initial.

Use:

Density = density\_previous x (100 - f) /100

### Case 2: Follow-up reduction cases for initial timestep in main costing loop

This case includes entries in the gis mapping file that have the **compartment, miu and nbal id** entries but only if the treatment is not initial.

Use:

Density = density\_previous x (100 - f) /100

### Case 3: Follow-up densify cases for initial timestep in main costing loop

This case includes entries in the gis mapping file that have only the **compartment and miu id**. It also includes the **compartment, miu and nbal id** entries but only if the treatment is initial.

Use:

Density = density\_previous x (100 + f) /100

### Case 5: Follow-up densify cases for initial timestep in main costing loop

This case includes entries in the gis mapping file that have the **compartment, miu and nbal id** entries but only if the treatment is not initial.

Use:

Density = density\_previous x (100 + f) /100

### Case 6: Follow-up densify cases for second and further timesteps in main costing loop

Note that the second and further timesteps are follow-up only.

This case includes all entries in the gis mapping file that are not **compartment** entries only, i.e. all valid entries.

Use:

Density = density\_previous x (100 + f) /100

## Worked Example

### Input: MIU Linked Species

These are two examples of the miu species in the linked species Excel file:

|  |  |  |  |
| --- | --- | --- | --- |
| M\_H60B400169 | Pinus pinaster | 30.000000 | Adult |
| M\_H60B400169 | Hakea sericea | 20.000000 | Young |
| M\_H60B400169 | Acacia longifolia | 5.000000 | Young |

|  |  |  |  |
| --- | --- | --- | --- |
| M\_H60B400176 | Pinus pinaster | 35.000000 | Young |
| M\_H60B400176 | Hakea sericea | 35.000000 | Young |
| M\_H60B400176 | Acacia longifolia | 20.000000 | Young |

The table below shows the explicit calculation of the density for all three scenarios, for initial, and follow up reduction.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Miu ID | Species | Idenscode | Initial reduction factor | Follow up reduction factor | Densify factor | E.g. initial reduction | E.g. follow-up reduction | E.g. densify |
| M\_H60B400169 | Pinus pinaster | 30 | 50 | 10 | 15 | 30\*(100-50)/100 = 15 | 30\*(100-10)/100 =27 | 30\*(100+15)/100 =34.5 |
| M\_H60B400169 | Hakea sericea | 20 | 50 | 0 | 10 | 20\*(100-50)/100 =10 | 20\*(100-0)/100 =20 | 20\*(100+10)/100 =22 |
| M\_H60B400169 | Acacia longifolia | 5 | 50 | 0 | 2 | 5\*(100-50)/100 =2.5 | 5\*(100-0)/100 =5 | 5\*(100+2)/100 =5.1 |
| M\_H60B400170 | Pinus pinaster | 25 | 50 | 10 | 12 | 25\*(100-50)/100 =12.5 | 25\*(100-10)/100 =22.5 | 25\*(100+12)/100 =28 |
| M\_H60B400170 | Acacia longifolia | 5 | 50 | 0 | 2.5 | 5\*(100-50)/100 =2.5 | 5\*(100-0)/100 =5 | 5\*(100+2.5)/100 =5.38 |
| 4.87M\_H60B400170 | Hakea sericea | 10 | 50 | 0 | 5 | 10\*(100-50)/100 =5 | 10\*(100-0)/100 =10 | 10\*(100+5)/100 =10.5 |

## Notes and Recommendations

\* **Algorithm optimization**: Note that this is a direct description of the algorithm, there are optimizations that should be applied which can then run the algorithm faster.

\* **Treatment Frequency**: High-frequency treatments accelerate density reduction, while low-frequency treatments delay reductions across years.

\* **Densification**: Accounts for ecological rebound of invasive species between treatments. Must always be applied at the end of each year for compartments that we not cleared.

\* **Reduction**: Accounts for clearing of invasive species between treatments (if budget was available and the compartment was a priority).

# Person Days

## Person Days Algorithm

The Person Days (PD) algorithm provides a standardized framework for estimating the labour requirements (in person-days) associated with clearing operations in compartments, management units (MIUs) and NBAL’s.

Note that this algorithm replaces earlier approaches that lacked slope dependence and that did not properly account for travel time.

The new algorithm introduces two tiers of calculation:

\* **Normal/Base Person Days** – a direct calculation based on Person Days variable PPD and the area of the compartment.

\* **Adjusted Person Days (or Person Days)** – an extended formulation that corrects the base values to account for travel overheads (walking and driving time) and slope effects.

This section describes both formulations, the underlying assumptions, and the computational workflow in detail.

## Normal Person Days (Base Calculation)

### Theory

Normal person days form the foundation of the calculation. They represent the idealized labour effort under standard working conditions, without explicit adjustment for terrain or travel time. The calculation is derived directly from the **Clearing Norms Table** where **PPD** values are prescribed for specific combinations of variables such as species growth form, treatment type, riparian/terrestrial classification, process stage (initial/follow-up), and density class.

The base person days for a compartment is expressed as:

PD\_base = PPD x Area

Where:

\* PPD = Person days variable (from clearing norms)

\* Area = Area of the compartment (ha)

#### Workflow

The workflow depends on the case:

* Initial timestep, initial or follow up
* First and further timesteps, as follow-up only

1. Iterate over the GIS mapping file: For each compartment, check whether there is only a compartment ID, or whether MIUs (Mapped Invaded Units) and/or NBAL records exist.

\* If **compartment only** (no MIU, no NBAL), no person days are assigned.

\* If MIU and/or NBAL are present, filter the associated linked species entries with preference to NBAL first.

2. Apply Subroutine 1 **(Treatment Method Selection)**:

\* If only one treatment method exists, select it directly.

\* If multiple treatment methods exist, apply ordered selection rules:

\* Adult + Landscape -> Ring bark

\* Adult + Riparian -> Fell

\* If unavailable -> Bark Strip

\* If still unavailable -> Lopping/Pruning

3. Apply Subroutine 2 **(Clearing Norm Filtering)**:

Filter the clearing norms spreadsheet step by step:

\* Match density (Idenscode or previous year density).

\* Match size class (Age).

\* Determine riparian/land classification.

\* Match species -> growth form -> validate against support growth form table.

\* Filter process (Initial vs Follow-up).

\* Apply treatment method filter.

\* At this stage, one valid record remains, if not use the first valid entry.

4. Extract PPD: The record provides the Person Days factor.

5. Compute base PD: Multiply PPD x Area.

## Adjusted Person Days

### Theory

The adjusted calculation incorporates real-world operational factors:

\* Walking and Driving Time Overheads: Time lost due to travel reduces effective clearing time.

\* Slope Dependence: Steeper slopes require proportionally greater labour effort.

Use this table to determine the slope range adjustment factors:

A screenshot of a computer

AI-generated content may be incorrect.

The adjustment is based on the following functional form:

PD\_adjusted = PD\_base x (WD) / ( (WD - 2 x (T\_walk + T\_drive)) / 60 )

Where:

\* PD\_ base = Base person days

\* WD = Standard working day length (default = 8 hours)

\* T\_walk, T\_drive = Walking and driving times (minutes)

\* The denominator represents effective working hours after subtracting travel overheads.

⚠️ Boundary conditions:

Finally, slope adjustments are applied:

PD\_final = PD\_adjusted x f\_slope

Where f\_slope is a multiplier derived from the slope class of the compartment.

#### Workflow

1. Obtain Base Person Days as described in the previous section.

2. Apply Adjustment for Travel Time:

\* Compute effective hours using walking + driving times.

\* Modify PD according to the formula above.

3. Apply Slope Adjustment

\* Look up slope range for the compartment.

\* Multiply adjusted PD by the slope factor.

4. Aggregate Across MIU-linked Species:

\* For each MIU species entry, compute adjusted PD.

\* Sum all adjusted PD values to get total per compartment.

## Example Calculation

### Example Inputs

\* Compartment ID: H\_123

\* Area = 15 ha

\* PPD = 0.35 person days factor (from clearing norms)

\* Working day (WD) = 8 hours

\* Walking time = 20 min

\* Driving time = 30 min

\* Slope factor = 1.4 (from slope of 25 degrees)

### Step 1: Base Person Days

PD\_base = 0.35 x 15 = 5.25

### Step 2: Travel Adjustment

PD\_adjusted = 5.25 x 8/ (8 - 2 x (20+30)/60) = 6.63

### Step 3: Slope Adjustment

PD\_adjusted = 6.63 x 1.4 = 9.28

### Result

\* Normal Person Days = 5.25

\* Adjusted Person Days = 9.28

## Notes and Recommendations

\* **Algorithm optimization**: Note that this is a direct description of the algorithm, there are optimizations that should be applied which can then run the algorithm faster.

\* **Algorithm upgrade:** Note that this algorithm replaces earlier approaches that lacked slope dependence and that did not properly account for travel time.

\* **Normal/Base Person Days** – a direct calculation based on Person Days variable PPD and the area of the compartment.

\* **Adjusted Person Days (or Person Days)** – an extended formulation that corrects the base values to account for travel overheads (walking and driving time) and slope effects.

# Costing

The costing algorithm provides a systematic approach for estimating the financial requirements of clearing operations across compartments and their associated MIU’s and NBAL’s. It integrates outputs from the density, prioritization, and person days algorithms with cost model parameters defined in the planning budgets.

This framework supports both **optimal budget scenarios** (unconstrained clearing) and **restricted budget scenarios** (user-defined yearly budget allocations), ensuring that the model can be applied to both planning and operational contexts.

## Parameters and Inputs

### Cost Model Parameters

Each costing model is defined by the following parameters (from the \*planning\_budgets\* spreadsheet in the MUCP input file):

\* Initial team size

\* Initial team cost/day

\* Follow-up team size

\* Follow-up team cost/day

\* Vehicle cost/day

\* Fuel cost/hour

\* Maintenance level

\* User-defined costs

### Supporting Data

To calculate costs, the following additional inputs are required:

\* MIU-linked species entries (from \*miu linked species\* spreadsheet)

\* Density per year (from the Density Algorithm)

\* Compartment priority scores (from the Prioritization Algorithm)

\* Person Days (both base and adjusted, from the Person Days Algorithm)

\* Drive time (from compartment shapefile)

\* Budget amount and yearly escalation percentage

## Algorithm Workflow

### Annual Iteration

The costing algorithm is applied sequentially from **Year 1 to the end year** (default: 20 years, but configurable).

## Preprocessing

1. Obtain the tree species data.

2. Obtain the **cost code** from the compartment shapefile.

3. Match this code to the \*planning\_budgets\* sheet to identify the **costing model name**.

4. Obtain the slope factor.

5. Calculate the priority (this can be done once) (see prioritization algorithm).

6. Obtain the treatment method per species.

7. Propagate the budgets over the timeframe of the simulation with simple interest.

## For year 1

## Step 1: Determine which entries are initial costing, and which are follow-up costing

Iterate through all compartments in the GIS mapping shapefile.

If there are NBAL’s with initial, and only compartment and MIU id entries in the gis mapping file, then these entries use the initial parameters. For any follow-ups, they will the follow-up parameters.

### Initial entries calculations

Step 1: First calculate the person days (see person day algorithm).

Step 2: Calculate the cost with initial parameters

Cost = PD\_adjusted x C\_init / S\_init + V\_day x PD\_base + U\_day x PD\_base

where:

( C\_init ) = initial team cost/day

( S\_init ) = initial team size (set to 1 if 0)

( V\_day ) = vehicle cost/day

( U\_day ) = user-defined cost/day

Step 3: Fuel Cost Distribution

For each compartment calculate the fuel cost using:

Fuel\_cost\_per\_compartment = F\_hour x T\_drive / 10

where ( **F\_hour** ) = fuel cost/hour and ( **T\_drive** ) = drive time (min).

Distribute fuel cost equally across MIU entries in the compartment.

Example: If you have 4 MIU entries for the compartment with costs {15, 20, 30, 35}, then if the fuel cost amounted to 40. Divide it equally i.e. each entry gets 40 / 4 = 10. The new cost per MIU entry is {25, 30, 40, 45} respectively.

Step 4: New density calculations

Use the density algorithm to calculate both the:

* **initial** reduction
* densification

Step 5: Flow calculations

Using the flow algorithm calculate the flow for the entries.

Step 6: Budget allocation and masking compartments cleared per budget

Before this can be done please do the follow-up clearing for year 1 and combine the results first. This in the next section

### Follow-up entries calculations

Step 1: First calculate the person days (see person day algorithm).

Step 2: Calculate the cost with initial parameters

Cost = PD\_adjusted x C\_follow / S\_follow + V\_day x PD\_base + U\_day x PD\_base

where:

( C\_follow ) = follow-up team cost/day

( S\_follow ) = follow-up team size (set to 1 if 0)

( V\_day ) = vehicle cost/day

( U\_day ) = user-defined cost/day

Step 3: Fuel Cost Distribution

For each compartment calculate the fuel cost using:

Fuel\_cost\_per\_compartment = F\_hour x T\_drive / 10

where ( **F\_hour** ) = fuel cost/hour and ( **T\_drive** ) = drive time (min).

Distribute fuel cost equally across MIU entries in the compartment.

Example: If you have 4 MIU entries for the compartment with costs {15, 20, 30, 35}, then if the fuel cost amounted to 40. Divide it equally i.e. each entry gets 40 / 4 = 10. The new cost per MIU entry is {25, 30, 40, 45} respectively.

Step 4: New density calculations

Use the density algorithm to calculate both the:

* **follow-up** reduction
* densification

Step 5: Flow calculations

Using the flow algorithm calculate the flow for the entries.

Step 6: Budget allocation and masking compartments cleared per budget

Combine the results of year 1 with the compartment (MIU and NBAL’s) entries here first.

Then obtain the current available budget for the year.

Then follow the algorithm for the budget distribution per compartment as in the below section.

For those entries that were cleared, a mask is applied to those entries, adjust the values as follows:

* prioritization does not change
* density, if cleared, use the initial or follow-up reduced density value, otherwise use the densified value
* person days, if cleared use the initial or follow-up person days, otherwise use 0
* flow, if cleared use the flow calculated value, otherwise use 0
* costing, if budget was available for clearing, use the cost calculated value, otherwise use 0

Each MIU/NBAL-linked species entry now has a **final cost allocation** for year 1.

## For year 2 onwards

Step 1: First calculate the person days (see person day algorithm).

Step 2: Calculate the cost with initial parameters

Cost = PD\_adjusted x C\_follow / S\_follow + V\_day x PD\_base + U\_day x PD\_base

where:

( C\_follow ) = follow-up team cost/day

( S\_follow ) = follow-up team size (set to 1 if 0)

( V\_day ) = vehicle cost/day

( U\_day ) = user-defined cost/day

Step 3: Fuel Cost Distribution

For each compartment calculate the fuel cost using:

Fuel\_cost\_per\_compartment = F\_hour x T\_drive / 10

where ( **F\_hour** ) = fuel cost/hour and ( **T\_drive** ) = drive time (min).

Distribute fuel cost equally across MIU entries in the compartment.

Example: If you have 4 MIU entries for the compartment with costs {15, 20, 30, 35}, then if the fuel cost amounted to 40. Divide it equally i.e. each entry gets 40 / 4 = 10. The new cost per MIU entry is {25, 30, 40, 45} respectively.

Step 4: New density calculations

Use the density algorithm to calculate both the:

* **follow-up** reduction
* densification

Step 5: Flow calculations

Using the flow algorithm calculate the flow for the entries.

Step 6: Budget allocation and masking compartments cleared per budget

Then obtain the current available budget for the year.

Then follow the algorithm for the budget distribution per compartment as in the below section.

For those entries that were cleared, a mask is applied to those entries, adjust the values as follows:

* prioritization does not change
* density, if cleared, use the follow-up reduced density value, otherwise use the densified value
* person days, if cleared use the follow-up person days, otherwise use 0
* flow, if cleared use the flow calculated value, otherwise use 0
* costing, if budget was available for clearing, use the cost calculated value, otherwise use 0

Each MIU/NBAL-linked species entry now has a **final cost allocation** for year 2 onwards.

## Budget Allocation after costing was calculated per compartment

The costing algorithm, before budget allocation assumes all compartments were cleared i.e. all costes were calculated to see what it would cost for the compartments to be cleared.

### Optimal Budget

Optimal budget means we have an unlimited budget.

\* All compartments are fully funded.

\* Costs are calculated and summed across MIU/NBAL-linked species per compartment.

\* Person days remain unadjusted.

\* Density reductions are applied uniformly.

The output is then that all compartments are costed.

### Constrained Budgets (Budgets 1–4)

When yearly budgets are limited:

1. Sort compartments by **priority** (descending).

2. Within equal priority: sort by **density (ascending)**, then **cost (ascending)**.

3. Iteratively allocate MIU entries (from highest to lowest priority):

\* Select lowest-cost, lowest-density entries first.

\* Deduct cost from available budget.

\* If subtraction results in budget < 0 -> skip entry, apply densification, set Person Days = 0.

4. Continue until budget is fully used.

## Budget Escalation

Yearly budgets are escalated using a **simple interest formulation:**

B\_y+1 = B\_y x (1+i)

Where:

\* ( B\_y ) = current year budget

\* ( i ) = yearly escalation percentage (decimal)

Budget leftovers are not carried forward.

## Assumptions

\* A 20-year planning horizon is used in the desktop previous version of the tool; currently the tool supports arbitrary durations making it more flexible for the user.

\* Simple yearly compounding is applied for budget escalation.

\* When a compartment’s cost reaches below the Maintenance level, the compartment is considered cleared.

\* Team sizes of 0 default to 1 for computational stability.

\* Budget leftovers are not carried forward.

## Example

### Year 1

Main cost for MIU/NBAL-linked species:

\* PD\_base = 10

\* PD\_adjusted = 12

\* Initial team size = 5

\* Initial team cost/day = R1000

\* Vehicle cost/day = R200

\* User-defined cost/day = R50

Cost = 12 x 1000 / 5 + 200 x 10 + 50 x 10 = 4900

Add fuel cost for compartment:

\* Fuel cost per hour = R100

\* Drive time = 60 minutes

Fuel\_cost\_per\_compartment = 100 x 60 / 10 = 600

Final cost for the compartment = **R4900 + R600 = R5500**.

### Year 2 onwards

Main cost for the MIU/NBAL-linked species:

\* PD\_base = 10

\* PD\_adjusted = 12

\* Follow-up team size = 20

\* Follow-up team cost/day = R500

\* Vehicle cost/day = R100

\* User-defined cost/day = R15

Cost = 12 x 500 / 20 + 100 x 10 + 15 x 10 = 1450

Add fuel cost for compartment:

\* Fuel cost per hour = R100

\* Drive time = 60 minutes

Fuel\_cost\_per\_compartment = 100 x 60 / 10 = 600

Final cost for the compartment = **R1450 + R600 = R2050**.

## Notes and Recommendations

\* **Algorithm optimization**: Note that this is a direct description of the algorithm, there are optimizations that should be applied which can then run the algorithm faster.

\* **Algorithm upgrade:** Note that this algorithm replaces earlier approaches. The costing is directly proportional to the person day calculation where the algorithm was corrected. This has a roll on effect to the costing algorithm.

# Flow

The flow algorithm is designed to quantify the reduction in mean annual runoff (MAR) attributable to invasive species within MIU’s and their associated NBALs. The algorithm integrates biological parameters (species attributes such as age class and density), hydrological parameters (MAR values, riparian versus dryland classification), and management rules (initial versus follow-up treatments) to produce compartment-level and species-level flow reductions over multiple years.

## Input Data Requirements

The following datasets are required:

\* **MIU and NBAL linked species file** (species composition and attributes).

\* **Species attribute table (Support data)** containing flow reduction parameters and density reduction factors:

- \*Flow Young\*

- \*Flow Seedling\*

- \*Flow Coppice\*

- \*Flow Optimal\*

- \*Flow Sub Optimal\*

- \*Initial Reduction\*

- \*Follow-up Reduction\*

\* **Prioritization file** containing compartment-level MAR values (mm/yr).

\* **Compartment shapefile attributes**, including:

\* \*density\*.

\* \*Area\* (hectares).

\* \*Riparian/Dryland classification\* (r or l).

\* **Density reduction factors** (initial and follow-up) as provided by the species composition support data.

## Algorithm Theory

The flow algorithm models the hydrological benefit (in terms of annual flow reduction) achieved through clearing invasive species. The key principle is that the **density of invasion multiplied by MAR** gives an estimate of the “potential water use” by invasive vegetation. This is adjusted through species-specific flow factors, treatment stages (initial vs. follow-up), and riparian multipliers.

The calculation proceeds in yearly iterations:

\* **Year 1** applies initial density reduction factors to all compartments, except those that were already a follow-up from year one already, these use the follow-up parameters for density reduction.

\* **Year 2 onward** applies follow-up density reduction factors.

\* Riparian units are assumed to have a **1.5x amplification factor** compared to dryland units.

## Algorithm Description

### ### Step A: Determine Flow Factor by Age Class

For each species entry linked to an MIU or NBAL:

\* **Mature / Adult / Mixed** -> Check growth condition, if sub-optimal, then use value from \*Flow Sub Optimal\* else use \*Flow Optimal\*.

\* **Young** -> Use value from \*Flow Young\* in Support file.

\* **Seedling** -> Use value from \*Flow Seedling\* in Support file.

\* **Coppice** -> Use value from \*Flow Coppice\* in Support file.

\* **Blank / Missing age class** ->Treat as Mature.

### ### Step B: Obtain the MAR per compartment, area and density

\* Obtain MAR (mm/yr) for compartment from prioritization file.

\* Extract density

\* Extract Area

### ### Step C: Year 1 calculation

**For already follow-up compartments:**

Compute the MAR density (for lack of a better name):

MAR\_density = Area x density x MAR

Flow Reduction Calculation (FR)

FR = MAR\_density x (100 – follow-up\_density\_reduction\_factor) / 100

For riparian compartments:

FR = FR x 1.5

Apply Flow Factor

FR = FR x Flow\_Factor

**For initial compartments:**

Compute the MAR density (for lack of a better name):

MAR\_density = Area x density x MAR

Flow Reduction Calculation (FR)

FR = MAR\_density x (100 – initial\_density\_reduction\_factor) / 100

For riparian compartments:

FR = FR x 1.5

Apply Flow Factor

FR = FR x Flow\_Factor

### ### Step D: Year 2 and further calculation

Compute the MAR density (for lack of a better name):

MAR\_density = Area x density x MAR

Flow Reduction Calculation (FR)

FR = MAR\_density x (100 – follow-up\_density\_reduction\_factor) / 100

For riparian compartments:

FR = FR x 1.5

Apply Flow Factor

FR = FR x Flow\_Factor

## Worked Example

Scenario:

Assume we are in follow-up. And this information:

\* MIU compartment, 100 ha area, density = 40, MAR = 800 mm/yr, Riparian = Yes.

\* Species entry: \*Young\* age class, Flow factor = 0.6, Initial density reduction = 30%, Follow-up = 10%.

**Step 1: Density of MAR**

MAR\_density = 100 x 40 x 800 = 3,600,000

**Step 2: Follow-up density reduction (10%)**

FR = 3,600,000 x (100 - 10) / 100 = 3,240,000

**Step 3: Riparian adjustment (×1.5)**

Is it riparian? Yes, it is.

FR = 3,240,000 x 1.5 = 4,860,000

**Step 4 (final): Flow factor adjustment (0.6)**

FR = 4,860,000 x 0.6 = 2,916,000

## Assumptions

1. **Age Classes**: Adult, Mature, and Mixed contribute zero flow reduction previously as per previous tool behaviour. The current algorithm uses flow optimal as per the algorithm received.

2. **Blank Age Classes**: Treated as Mature (0 flow factor previously, not flow optimal or flow sub-optimal).

3. **Riparian Adjustment**: Uniform multiplier of 1.5 applied to dryland FR values.

4. **Density of MAR**: A derived term introduced for consistency; not a hydrological standard.

5. **Yearly Iteration**: Assumes sequential application of follow-up density reductions after Year 1.

6. **Missing MAR Values**: If no MAR is provided for a unit in the prioritization file, all flow calculations default to zero.

## Notes and Recommendations

\* **Algorithm optimization**: Note that this is a direct description of the algorithm, there are optimizations that should be applied which can then run the algorithm faster.

\* **Algorithm upgrade:** Note that this algorithm replaces earlier approaches that lacked riparian adjustment and the selection of the flow factor which did not properly account for the mature/adult/mixed category.

# Main costing loop

The **main costing loop** is the central part of the MUCP code engine that integrates all the algorithms including prioritization, person-day calculations, costing, density updates, and flow reduction calculations across compartments, MIU’s, and NBAL’s entries. This loop is executed for each time step in the simulation and forms the backbone of the budget allocation and cost optimization framework.

## Initial Time Step (Year 1)

At the first time step, the GIS mapping data is split into **three distinct categories** to ensure correct initialization of costs and densities:

1. **Compartment ID only and maintenance compartments**

\* These entries are **left unchanged** during the initial calculation loop.

\* They represent compartments already allocated to maintenance or compartments that have no linked MIU/NBAL identifiers.

2. **NBAL entries with initial treatments** and GIS mapping entries that contain a **compartment ID and MIU ID only** (no NBAL ID)

\* These are processed as **initial treatments**.

\* Costing is performed using **initial parameters** such as the \*Initial Team Size\* and \*Initial Cost/Day\* from the costing model.

3. **Entries with compartment ID, MIU ID, and NBAL ID**

\* These represent **follow-up treatments**.

\* Costing is performed using the **follow-up parameters**, e.g., \*Follow-up Team Size\* and \*Follow-up Cost/Day\*.

This separation ensures that Year 1 differentiates between untouched compartments, compartments requiring initial clearing, and those already undergoing follow-up interventions.

## Subsequent Time Steps (Year 2 onwards)

From Year 2 and beyond:

\* The **three categories above are combined** into a single pool of entries.

\* All entries are treated as **follow-up operations**, regardless of whether they were initial or follow-up in Year 1.

\* Follow-up costing parameters are consistently applied for the remaining years of the simulation.

This reflects the ecological and operational reality that once a compartment is initially treated, all further interventions are follow-up actions.

## Budget Optimization and Masking

Before executing the main costing loop in each time step:

1. The **optimal budget** for that year is calculated based on input budget plans (e.g., 10,000,000 ZAR, 7,500,000 ZAR, etc.).

2. A **mask (‘cleared\_now’)** is applied:

\* The annual budget is divided across compartments according to prioritization scores.

\* Only compartments that can be cleared within the available budget are selected.

\* These compartments are marked as ‘cleared\_now’ and included in the budget plan for that year.

\* Compartments that cannot be cleared due to insufficient budget are deferred to subsequent years.

## Data Preparation

Before the loop executes, the following user supplied datasets are prepared and validated:

\* **Compartment Prioritization** (from prioritization CSV)

\* **Compartment geometries and attributes** (area, slope, drive/walk time, costing level, etc.)

\* **MIU geometries and riparian classification (l/r)**

\* **NBAL geometries and stage**

\* **Linked species attributes** (from MIU/NBAL linked species files)

\* **Species attributes** (growth form, reductions, flow factors, densification, treatment frequencies, etc.)

\* **Clearing norms** (person-days per density, growth form, and method)

\* **Costing data** (team sizes, daily rates, escalation, vehicle/fuel/maintenance costs, etc.)

This ensures that the prioritization, density, flow, and costing calculations can be executed in the correct order.

## Calculation Sequence Within Each Time Step

For each selected compartment in the timestep, the following ordered sequence is executed:

1. **Person-Day Calculation**

\* Determined using compartment slope, terrain type, growth form, and clearing norms.

\* Adjusted by treatment type (initial or follow-up).

2. **Costing Calculation**

\* Person-days are multiplied by the relevant costing parameters (team size, daily rates, escalation factors).

\* Additional overhead costs (vehicle, fuel, maintenance) are added.

3. **Density Update**

\* Density values are reduced based on the initial or follow-up reduction factors from the species attributes.

\* Densification (if applicable) is applied to simulate regrowth dynamics.

4. **Flow Calculation (if MAR available)**

\* Flow reduction is calculated per species and aggregated to unit-level reductions.

\* If MAR is missing, flow reductions default to 0.

This calculation order is critical:

\* Person-days must be established before costing.

\* Costing must be finalized before density updates.

\* Density updates must precede flow calculations.

## Example Input Data Structures

The following input datasets are required for the costing loop. These are loaded both in **viewer** (user interface, i.e. support data) and **user** (user input files) contexts.

### GIS Mapping, NBAL, MIU and compartment shapefiles

**Gis mapping shapefile has the following structure:**

Index(['nbal\_id', 'miu\_id', 'compt\_id', 'area', 'geometry'], dtype='object')

nbal\_id h60b400052

miu\_id m\_h60b400052

compt\_id c\_h60b400303

area 0.000008

geometry POLYGON ((21496.64963930845 -3754318.10992603,...

Name: 0, dtype: object

**NBAL shapefile has the following structure:**

Index(['nbal\_id', 'area', 'geometry', 'stage'], dtype='object')

nbal\_id h60b400177

area 33.991674

geometry POLYGON ((11084.678312947115 -3760290.90121776...

stage 5

Name: 0, dtype: object

**MIU shapefile has the following structure:**

Index(['miu\_id', 'area', 'riparian\_c', 'geometry'], dtype='object')

miu\_id m\_h60b400169

area 57.345913

riparian\_c l

geometry POLYGON ((9582.974294258824 -3761027.339255519...

Name: 0, dtype: object

**Compartments shapefile has the following structure:**

Index(['compt\_id', 'area\_ha', 'slope', 'walk\_time', 'drive\_time', 'costing',

'grow\_con', 'geometry'],

dtype='object')

compt\_id c\_h60b400177

area\_ha 33.991674

slope 28.075625

walk\_time 35.0

drive\_time 30.0

costing 2

grow\_con sub-optimal

geometry POLYGON ((11084.678312947115 -3760290.90121776...

Name: 0, dtype: object

### MIU and NBAL Linked Species Excel files

**MIU linked species file has the following structure:**

Index(['miu\_id', 'species', 'idenscode', 'age'], dtype='object')

miu\_id m\_h60b400052

species pinus pinaster

idenscode 35.0

age young

Name: 0, dtype: object

**NBAL linked species file has the following structure:**

Index(['nbal\_id', 'species', 'idenscode', 'age'], dtype='object')

nbal\_id h60b400052

species pinus pinaster

idenscode 35.0

age young

Name: 0, dtype: object

### Compartment Priorities CSV file

**Compartment prioritization file has the following structure:**

Index(['compt\_id', 'owner', 'vegetation status', 'veld age', 'elevation',

'rain', 'runoff'],

dtype='object')

compt\_id c\_h60b400169

owner capenature

vegetation status lt

veld age 8

elevation 806

rain 1195.3

runoff 598

Name: 0, dtype: object

### Species Attributes (support data)

Index(['id', 'species\_name', 'genus', 'english\_name', 'afrikaans\_name',

'growth\_form', 'wc', 'nc', 'kzn', 'gtg', 'mpl', 'fs', 'ec', 'lmp', 'nw',

'initial\_reduction', 'follow\_up\_reduction', 'treatment\_frequency',

'densification', 'flow\_optimal', 'flow\_sub\_optimal', 'flow\_young',

'flow\_seedling', 'flow\_coppice'],

dtype='object')

id 17

species\_name acacia longifolia

genus acacia

english\_name long-leaved wattle

afrikaans\_name langblaarwattel

growth\_form sprouting tree

wc 1

nc 1

kzn 1

gtg 1

mpl 1

fs 1

ec 1

lmp 0

nw 0

initial\_reduction 50

follow\_up\_reduction 20

treatment\_frequency 3

densification 15

flow\_optimal 0.9

flow\_sub\_optimal 0.9

flow\_young 0.5

flow\_seedling 0.1

flow\_coppice 0.5

Name: 16, dtype: object

### Costing Data (support data)

Index(['Costing Model Name', 'Initial Team Size', 'Initial Cost/Day',

'Follow-up Team Size', 'Follow-up Cost/Day', 'Vehicle Cost/Day',

'Fuel Cost/Hour', 'Maintenance Level', 'Cost/Day'],

dtype='object')

Costing Model Name mumba

Initial Team Size 1

Initial Cost/Day 60.00

Follow-up Team Size 2

Follow-up Cost/Day 3.00

Vehicle Cost/Day 3.00

Fuel Cost/Hour 3.00

Maintenance Level 2

Cost/Day 50.00

Name: 0, dtype: object

# Release notes & references

## Release notes

The open-source implementation of the MUCP tool introduces several important differences compared to the previous (proprietary) version. These differences affect both the underlying algorithms and the functionality available to users.

### Algorithmic Differences

#### Prioritization and Density

The implementations of the prioritization and density remain consistent with the previous tool. This ensures continuity of results where these parameters are concerned, so outputs based on these algorithms are directly comparable between the two systems.

#### Person-Days Calculation

The person-days algorithm has undergone a significant change.

In the previous tool, the calculation did not account for slope and deviated from the method provided in the official Person Days Excel spreadsheet provided by DFFE.

In the open-source tool, the algorithm has been corrected and now explicitly incorporates slope dependence. This results in more realistic estimates of the effort required, especially in areas with non-trivial terrain gradients.

#### Costing Algorithm

The costing (budgeting) algorithm continues to use prioritization, density, and person-days as the core inputs. However:

Because the person-days calculation has been corrected, the overall costing outputs will differ from the previous tool.

Depending on compartment-specific parameters such as slope, area, and effort requirements, costs may be higher or lower than those reported by the previous version.

This adjustment results in outputs that are better aligned with real-world field conditions and financial planning.

#### Flow algorithm

The flow algorithm underwent a slight change from the previous tool which did not have a riparian factor adjustment present. The riparian depending if it is landscape or riparian, needed a factor of 1.5 adjustment. This is now incorporated into the new algorithm set.

This means that the flow results from the previous tool will be different than the current opensource tool. Note also that the flow would have changed due to the person day algorithm being updated with the slope factor correction.

### Algorithm Derivation Methodology

The algorithms used in the opensource version were mainly reverse-engineered through brute-force testing of the desktop tool. This involved painstaking sensitivity analyses, where individual input values were varied systematically, and the resulting outputs were analysed to infer the governing equations.

As a result, the algorithmic logic was reconstructed based on observed tool behaviour rather than pre-existing documentation.

While the derived algorithms reproduce the functional behaviour of the original tool with a high degree of fidelity, there may be minor differences due to the lack of direct access to the original formulations.

### Functional and Structural Differences

#### User-Defined Prioritization Categories

The previous tool limited users to a fixed set of prioritization categories. The open-source version removes this limitation, allowing users to define their own prioritization variables and categories. This provides greater flexibility for tailoring the tool to local or project-specific management needs.

#### Extended Budget Horizons

Budgets are no longer restricted to a 20-year planning horizon. Users can project costs and treatment plans for arbitrary timeframes, supporting both long-term strategic planning and shorter adaptive management cycles.

#### Herbicides and Cost Recovery

As in the previous version, herbicide usage and cost recovery mechanisms have not yet been implemented. These remain placeholders for potential future enhancements.

#### Corrections to Column Names, Units, and Terminology

Several column names, units, and descriptive terms from the previous tool were found to be inconsistent or unclear. These have been corrected and standardized in the open-source implementation, ensuring greater clarity and consistency of outputs.

### Technical Implementation

#### Open-Source Codebase

The tool has been re-implemented in Python and is fully open-source.

The algorithmic core is implemented primarily using the Pandas package for high-performance data manipulation and calculation.

The user interface and visualization components are delivered through a Django-based web application, supported by auxiliary Python packages.

This separation of concerns between algorithms and the web viewer ensures modularity and makes it easier to extend, maintain, and audit the system.

### Beta Release

This release should be considered beta software. While the tool is functional and provides scientifically valid outputs, there may still be bugs or edge cases that need to be resolved. User feedback during this stage is critical to refine and stabilize the system.

## References

Some references that were useful in figuring out the MUCP algorithms.

* <https://scholar.sun.ac.za/bitstream/handle/10019.1/123841/boast_cost_2021.pdf?sequence=2&isAllowed=y>
* <https://sites.google.com/site/wfwplanning/planning>
* Greg Forsyth’s manual found in the link above
* Andrew Wannenburgh’s Person Days Excel spreadsheet
* David le Maitre’s email with the Flow algorithm and riparian 1.5 factor adjustment correction

Manual End