

---

# **MAUD Documentation**

***Release 0.8***

**Guilherme Castelão & Bia Villas Boas**

July 04, 2016



|          |   |          |
|----------|---|----------|
| <b>1</b> | <b>Welcome to MAUD's documentation!</b> | <b>1</b> |
| 1.1      | Installation . . . . .                  | 1        |
| 1.2      | Getting Started with MAUD . . . . .     | 1        |
| <b>2</b> | <b>Indices and tables</b>               | <b>3</b> |



---

## Welcome to MAUD's documentation!

---

MAUD stands for Moving Averages for Uneven Data. The major characteristic of this package is to expect an uneven distribution of the data. In the case of regular geographic grid, i.e. regular latitudes and longitudes, the weight windows will be defined based on the distance.

Contents:

## 1.1 Installation

### 1.1.1 Requirements

- Python 2.6 ( $\geq 2.6.5$ ), 2.7, 3.1 or 3.2
- Numpy
- distribute 0.6 ( $\geq 0.6.40$ )

### 1.1.2 Installing MAUD

#### Using pip

```
pip install --no-deps maud
```

---

**Note:** The `--no-deps` flag is optional, but highly recommended if you already have Numpy installed, since otherwise pip will sometimes try to “help” you by upgrading your Numpy installation, which may not always be desired.

---

## 1.2 Getting Started with MAUD

One can use MAUD inside Python or in the shell.

### 1.2.1 Inside Python

```
>>> from maud import window_1Dmean, window_mean_2D_latlon
>>> window_1Dmean(x, l=200e3, t=None, method='hann', axis=0, parallel=True)
```

```
>>> window_mean_2D_latlon(Lat, Lon, data, l, method='hamming', interp=False)
```

### The faster version

There is a Cython version of each filter. If you're able to, use `cmaud` instead of `maud` to gain at least one order of magnitude on the speed.

```
>>> from cmaud import window_1Dmean
>>> window_1Dmean(x, l=200e3)
```

```
>>> from cmaud import window_mean_2D_latlon
>>> window_mean_2D_latlon(Lat, Lon, data, l)
```

### Shell script

To check the available options

```
>>> maud4latlonnc -h
```

On this example, it will filter the variable 'temperature' at the netCDF file `model_output.nc`. The weight function used is hamming, and the window scale is `600e3`. The `npes` define the number of parallel process to be used, in this case 18. The option `-interp` defines that any masked value will be replaced in the output as the filtered result of the valid values around it, inside the window length.

```
>>> maud4latlonnc -l 600e3 --var='temperature' -w hamming --interp --npes=18 model_output.nc
```

---

## Indices and tables

---

- `genindex`
- `modindex`
- `search`