e-6

March 11, 2025

[2]:
```
import numpy as np
from numpy import ma
```

[3]:
```
"""
Rationale
Masked arrays are arrays that may have missing or invalid entries.
The numpy.ma module provides a nearly work-alike replacement for numpy that␣
 ↪supports data arrays with masks.

What is a masked array?
In many circumstances, datasets can be incomplete or tainted by the presence of␣
 ↪invalid data.
For example, a sensor may have failed to record a data, or recorded an invalid␣
 ↪value.
The numpy.ma module provides a convenient way to address this issue, by␣
 ↪introducing masked arrays.
A masked array is the combination of a standard numpy.ndarray and a mask.
A mask is either nomask, indicating that no value of the associated array is␣
 ↪invalid, or
an array of booleans that determines for each element of the associated array␣
 ↪whether the value is
valid or not. When an element of the mask is False,
the corresponding element of the associated array is valid and is said to be␣
 ↪unmasked.
When an element of the mask is True, the corresponding element of the␣
 ↪associated array is said to
be masked (invalid).

The package ensures that masked entries are not used in computations.
"""
```

[3]:
```
'\nRationale\nMasked arrays are arrays that may have missing or invalid
entries.\nThe numpy.ma module provides a nearly work-alike replacement for numpy
that supports data arrays with masks.\n\nWhat is a masked array?\nIn many
circumstances, datasets can be incomplete or tainted by the presence of invalid
data.\nFor example, a sensor may have failed to record a data, or recorded an
invalid value.\nThe numpy.ma module provides a convenient way to address this
```

```
issue, by introducing masked arrays.\nA masked array is the combination of a
standard numpy.ndarray and a mask.\nA mask is either nomask, indicating that no
value of the associated array is invalid, or\nan array of booleans that
determines for each element of the associated array whether the value is \nvalid
or not. When an element of the mask is False, \nthe corresponding element of the
associated array is valid and is said to be unmasked.\nWhen an element of the
mask is True, the corresponding element of the associated array is said to \nbe
masked (invalid).\n\nThe package ensures that masked entries are not used in
computations.\n'
```

[4]:
```
a = np.arange(-3, 3)
a
```

[4]:
```
array([-3, -2, -1,  0,  1,  2])
```

[5]:
```
b = np.array([
    [1, 2, 3],
    [4, np.nan, 6]
])
5
```

[5]:
```
5
```

[8]:
```
m = ma.masked_array(b, mask=[0, 0, 0, 0, 1, 0])
m
```

[8]:
```
masked_array(
  data=[[1.0, 2.0, 3.0],
        [4.0, --, 6.0]],
  mask=[[False, False, False],
        [False,  True, False]],
  fill_value=1e+20)
```

[9]:
```
m_2 = ma.masked_invalid(b)
m_2
```

[9]:
```
masked_array(
  data=[[1.0, 2.0, 3.0],
        [4.0, --, 6.0]],
  mask=[[False, False, False],
        [False,  True, False]],
  fill_value=1e+20)
```

[10]:
```
m_3 = ma.masked_where(a <= 0, a)
m_3
```

[10]:
```
masked_array(data=[--, --, --, --, 1, 2],
             mask=[ True,  True,  True,  True, False, False],
       fill_value=999999)
```

[11]:
```
m_4 = ma.masked_values(a, -1)
m_4
```

[11]:
```
masked_array(data=[-3, -2, --, 0, 1, 2],
             mask=[False, False,  True, False, False, False],
       fill_value=-1)
```

[ ]:
```

```