Python Documentation Numpy Doc

Fan Wang

2020-10-21

Contents

1 Numpy Doc Documentation Guide

Go to the **RMD**, **PDF**, or **HTML** version of this file. Go back to Python Code Examples Repository (bookdown site) or the pyfan Package (API).

- sphinxcontrib-napoleon examples.
- numpydoc examples.
- Documenting Python APIs with docstrings
- Numpy Doc Example
-

1.1 Parameters

Check types:

```
print(type(111))
print(type('111'))
import logging
print(type(logging.WARNING))
```

Style 1:

```
Parameters
----------
n : int
    The upper limit of the range to generate, from 0 to `n` - 1.
param1 : int
    The first parameter.
param1 : str
    Description of `param1`.
msg : str
    Human readable string describing the exception.
param1 : int
```

```
    The first parameter.
param2 : str
    The second parameter.
param3 : str, optional
    The second parameter.
param5: dict
    A dictionary
param6: bool
    boolean
arr1 : ndarray
    2D array containing data with `float` type.
arr2 : ndarray
    1D mask array(containing data with boolean type).
```

Style 2, this will add a link to the types in python doc:

```
Parameters
----------
param2 : :obj:`str`, optional
    The second parameter.
code : :obj:`int`, optional
    Numeric error code.
param3 : :obj:`int`, optional
    Description of `param3`.
param4 : :obj:`list` of :obj:`str`
    Description of `param2`. Multiple
    lines are supported.
```

For args and kwargs:

```
Parameters
----------
*args
    Variable length argument list.
**kwargs
    Arbitrary keyword arguments.
```

1.2 Returns

```
Returns
-------
numpy.array of shape (1, it_draws)
    A vector of sorted or unsorted random grid points, or equi-quantile
    points.

Returns
-------
:obj:`tuple` of :obj:`bool`
```

```
Returns
-------
None
```

1.3 Function Calls

To refer to functions in the same .py file, just need to use: *:func:`log_format`* to refer to function name. For function in different .py files, might need its full path

```
**kwargs
  Arguments for functions that is called, including :func:`log_format`
```

1.4 Examples

Array outputs.

```
Examples
--------
>>> fl_mu = 0
>>> fl_sd = 1
>>> it_draws = 5
>>> it_seed = 123
>>> fl_lower_sd = -1
>>> fl_higher_sd = 0.8
>>> it_draw_type = 0
>>> ar_draw_random_normal(fl_mu, fl_sd, it_draws,
...                       it_seed, it_draw_type,
...                       fl_lower_sd, fl_higher_sd)
[-1.          0.8         0.2829785 - 1. - 0.57860025]
>>> it_draw_type = 1
>>> ar_draw_random_normal(fl_mu, fl_sd, it_draws,
...                       it_seed, it_draw_type,
...                       fl_lower_sd, fl_higher_sd)
[-1. - 0.47883617 - 0.06672597  0.3338994   0.8]
>>> it_draw_type = 2
>>> ar_draw_random_normal(fl_mu, fl_sd, it_draws,
...                       it_seed, it_draw_type,
...                       fl_lower_sd, fl_higher_sd)
[-1. - 1. - 0.57860025  0.2829785   0.8]
```

String outputs.

```
Examples
--------
>>> log_vig_start(spt_root = proj_sys_sup.main_directory(),
...               main_folder_name='logvig', sub_folder_name='parameters',
...               subsub_folder_name='combo_type',
...               file_name='fs_gen_combo_type',
...               it_time_format=8, log_level=logging.INFO)
C:\\Users\\fan\\logvig\\parameters\\combo_type\\fs_gen_combo_type_20201030.log.py
```