PyWinda

Release 1.2.2

PyWind Authors

May 22, 2022

CONTENTS:

PyWinda library is developed to simplify the power and load calculations at a wind farm level.

- For a complete documentation click here!

If you want to contribute to PyWinda library:

- For a guide to contribution click here!
- To fork and make your contribution in Github click here!

CHAPTER

ONE

PYWINDA

1.1 Introduction

In PyWinda library there are currently two modules which are maintained regularly. The pywinda (written in lower case) is the main module which creates a windfarm. The second module is dfm_module, which does all the reliability calculations.

In pywinda, there are two types of supported wind turbines, the single rotor turbiens (SRT) and multirotor turbines (MRT). It is always necessary to make a wind farm in pywinda first. Later you can add more turbines and also add an environment to the already created wind farm. Note that you can not create a standalone wind turbine without assigning it to a wind farm. However, you can make several standalone environments and assign them to the wind farms one at a time.

1.2 pywinda module

class pywinda.**MRT**(*srtUniqueID*, *diameter=nan*, *hubHeigt=nan*, *x_horizontal=nan*, *y_vertical=nan*, *ws=[]*, *cp=[]*)

Bases: *pywinda.SRT*

Inherited from the class SRT, all the methods of a SRT is available for the MRT as well. The part is under development.

class pywinda.**SRT**(*srtUniqueID*, *diameter=nan*, *hubHeigt=nan*, *x_horizontal=nan*, *y_vertical=nan*, *ws=[]*, *cp=[]*)

Bases: `object`

Creates single rotor turbine (SRT) object and returns it with the given unique name.

Parameters

- **srtUniqueID** – [*req*] Unique Id of the wind turbine as a string.
- **diameter** – [*opt*] diameter of the SRT.
- **hubHeight** – [*opt*] hub height of the SRT.
- **x_horizontal** – [*opt*] x coordinate of the SRT.
- **y_vertical** – [*opt*] y coordinate of the SRT.

Example

```
>>> WT1=SRT("TheWT1",diameter=150)
>>> print(WT1.info)
       Property                                                Value
0   Unique Name                                              TheWT1
1  x_horizontal                                                 NaN
2    y_vertical                                                 NaN
3      Diameter                                                 150
4    Hub height                                                 NaN
5          Area                                        17671.458676
6    Windspeeds  [0.0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, ...
7            CP                                                  []
```

---

property info

Returns a data frame containing information about the wind turbine.

Parameters **None** –

Example

```
>>> from PyWinda import pywinda as pw
>>> Curslack = pw.windfarm("Curslack_farm")
>>> WT1 = Curslack.addTurbine("C_WT1", hubHeigt=120, diameter=120)
>>> print(WT1.info)
       Property                                                Value
0   Unique Name                                               C_WT1
1  x_horizontal                                                 NaN
2    y_vertical                                                 NaN
3      Diameter                                                 120
4    Hub height                                                 120
5          Area                                        11309.733553
6    Windspeeds  [0.0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, ...
7            CP                                                  []
```

---

class pywinda.**environment**(*uniqueID*)

Bases: `object`

Creates the stand-alone environment and returns it with the given unique ID. By default, wind speeds from 0 m/s to 50 m/s with an increment of 0.5 m/s and also 360 degree with 1 degree increment are also added. The temperature of 25 degree Celsius and pressure of 101325 Pa is assumed. See example below:

Parameters **uniqueID** – [*req*] the given unique ID.

Example

```
>>> Env = environment("C_Env")
>>> #Creates an environment without assigning it to any wind farm.
>>> print(Env.info.keys())
dict_keys(['Wind directions', 'Sectors', 'Wind speeds', 'Pressure',
↪'Temperature', 'Wind probability', 'Scale parameter of wind␣
↪distribution', 'Shape parameter of wind distribution'])
```

(continues on next page)

(continued from previous page)

```
>>> print(Env.info['Wind directions'])
[0, 1, 2, 3, ...]
>>> print(Env.info['Wind speeds'])
[0.0, 0.5, 1.0, 1.5, 2.0, ...]
```

---

property info

Returns all the defined conditions of the environment.

Parameters **None** –

Example

```
>>> dantysk=windfarm("DanTysk")
>>> env=environment("D_Env")
>>> print(env.info.keys())
dict_keys(['Wind directions', 'Sectors', 'Wind speeds', 'Pressure',
↪'Temperature', 'Wind probability', 'Scale parameter of wind␣
↪distribution', 'Shape parameter of wind distribution'])
>>> print(env.info['Wind directions'])
[0, 1, 2, 3, ...]
>>> print(env.info['Wind speeds'])
[0.0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, ...]
```

---

makeSectors(*n=12*, *sectorNames=['N_0', 'NNE_30', 'NEN_60', 'E_90', 'ESE_120', 'SSE_150', 'S_180', 'SSW_210', 'WSW_240', 'W_270', 'WNW_300', 'NNW_330']*)

Creates the given sectors to the related environment. Returns the result as a data frame. Divides the 360 degrees to given number of sectors. By default it divides to 12 sectors and assigns the 12 standard names for every sector e.g. N_0 starts from 346 degrees and ends at 15 degrees.

Parameters

- **n** – [*opt*] the number of sectors.
- **sectorNames** – [*opt*] names of the sectors given by user or default names for n=12.

Example

```
>>> Env=environment("C_Env2")
>>> print(Env.makeSectors())
      N_0  NNE_30  NEN_60   E_90  ...  WSW_240  W_270  WNW_300  NNW_
↪330
0   346.0    16.0    46.0   76.0  ...    226.0  256.0    286.0    316.
↪0
1   347.0    17.0    47.0   77.0  ...    227.0  257.0    287.0    317.
↪0
2   348.0    18.0    48.0   78.0  ...    228.0  258.0    288.0    318.
↪0
3   349.0    19.0    49.0   79.0  ...    229.0  259.0    289.0    319.
↪0
```

(continues on next page)

(continued from previous page)

```
4   350.0    20.0    50.0   80.0  ...    230.0  260.0    290.0    320.
↪0
5   351.0    21.0    51.0   81.0  ...    231.0  261.0    291.0    321.
↪0
6   352.0    22.0    52.0   82.0  ...    232.0  262.0    292.0    322.
↪0
7   353.0    23.0    53.0   83.0  ...    233.0  263.0    293.0    323.
↪0
8   354.0    24.0    54.0   84.0  ...    234.0  264.0    294.0    324.
↪0
9   355.0    25.0    55.0   85.0  ...    235.0  265.0    295.0    325.
↪0
10  356.0    26.0    56.0   86.0  ...    236.0  266.0    296.0    326.
↪0
11  357.0    27.0    57.0   87.0  ...    237.0  267.0    297.0    327.
↪0
12  358.0    28.0    58.0   88.0  ...    238.0  268.0    298.0    328.
↪0
13  359.0    29.0    59.0   89.0  ...    239.0  269.0    299.0    329.
↪0
14    0.0    30.0    60.0   90.0  ...    240.0  270.0    300.0    330.
↪0
15    1.0    31.0    61.0   91.0  ...    241.0  271.0    301.0    331.
↪0
16    2.0    32.0    62.0   92.0  ...    242.0  272.0    302.0    332.
↪0
17    3.0    33.0    63.0   93.0  ...    243.0  273.0    303.0    333.
↪0
18    4.0    34.0    64.0   94.0  ...    244.0  274.0    304.0    334.
↪0
19    5.0    35.0    65.0   95.0  ...    245.0  275.0    305.0    335.
↪0
20    6.0    36.0    66.0   96.0  ...    246.0  276.0    306.0    336.
↪0
21    7.0    37.0    67.0   97.0  ...    247.0  277.0    307.0    337.
↪0
22    8.0    38.0    68.0   98.0  ...    248.0  278.0    308.0    338.
↪0
23    9.0    39.0    69.0   99.0  ...    249.0  279.0    309.0    339.
↪0
24   10.0    40.0    70.0  100.0  ...    250.0  280.0    310.0    340.
↪0
25   11.0    41.0    71.0  101.0  ...    251.0  281.0    311.0    341.
↪0
26   12.0    42.0    72.0  102.0  ...    252.0  282.0    312.0    342.
↪0
27   13.0    43.0    73.0  103.0  ...    253.0  283.0    313.0    343.
↪0
28   14.0    44.0    74.0  104.0  ...    254.0  284.0    314.0    344.
↪0
29   15.0    45.0    75.0  105.0  ...    255.0  285.0    315.0    345.
↪0
```

(continues on next page)

(continued from previous page)

```
[30 rows x 12 columns]
```

---

windConditions(*windProbability=[0.08333333333333333, 0.08333333333333333, 0.08333333333333333, 0.08333333333333333, 0.08333333333333333, 0.08333333333333333, 0.08333333333333333, 0.08333333333333333, 0.08333333333333333, 0.08333333333333333, 0.08333333333333333, 0.08333333333333333]*, *aParams=[7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7]*, *kParams=[2.5, 2.5, 2.5, 2.5, 2.5, 2.5, 2.5, 2.5, 2.5, 2.5, 2.5, 2.5]*)

Creates and assigns the given wind conditions to the related environment. Returns the result as a data frame. Divides the 360 degrees to given number of sectors. By default it divides to 12 sectors and assigns the 12 standard names for every sector e.g. N_0 starts from 346 degrees and ends at 15 degrees.

param windProbability [*opt*] the probabiliyt of wind presence in each sector, by default equal to 1/12.

param aParams [*opt*] the scale factor of the weibull distribution of the wind in the sector, by default equal to 7 m/s .

param kParams [*opt*] the shape factor of the weibull distribution of the wind int the sector, by default equla to 2.5.

Example

```
>>> from PyWinda import pywinda as pw
>>> Env=pw.environment("C_Env2")
```

---

class pywinda.**windfarm**(*uniqueID*, *lifetime=788400000*)

Bases: `object`

Creates wind farm object with the given unique ID. Pywinda will also create an internal shallow copy of the same windfarm object.

Parameters **uniqueID** – [*req*] Unique Id of the wind farm as a string.

Example

```
>>> from PyWinda import pywinda as pw
>>> curslack = pw.windfarm("Curslack_uID")
>>> print(pw.Curslack_uID==curslack)
True
```

---

addRefTurbine(*uniqueID*, *reference='NREL'*)

By default adds a single rotor turbine (SRT) rference turbine to the related windfarm. Returns the created wind turbine with the given unique ID. The wind turbine would be callable via its unique name and via the assigned variable by user. Note that the referenced unique id is stored in library. Thus when calling the turbine via unique id, it should be prefixed by library name pywinda. See example below.

Parameters

- **uniqueID** – [*req*] Unique ID of the wind turbine as string
- **reference** – [*opt*] Choose among 'NREL-5MW' or 'DTU-10MW' reference turbines

Example

```
>>> from PyWinda import pywinda as pw
>>> DanTysk=pw.windfarm('DanTysk01')
>>> WT1=DanTysk.addRefTurbine('Turbine1',reference='NREL')
>>> print(pw.Turbine1.info)
       Property                                                  Value
0   Unique Name                                               Turbine1
1  x_horizontal                                                    NaN
2    y_vertical                                                    NaN
3      Diameter                                                    120
4    Hub height                                                    120
5          Area                                           11309.733553
6    Windspeeds  [0.0, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, ...
7            CP  [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.198, 0.313, 0...
```

---

addTurbine(*uniqueID*, *turbineType='SRT'*, *diameter=nan*, *hubHeigt=nan*, *x_horizontal=nan*, *y_vertical=nan*, *ws=[]*, *cp=[]*)

By default adds a single rotor turbine (SRT) to the related windfarm. Returns the created wind turbine with the given unique ID. The wind turbine would be callable via its unique name and via the assigned variable by user. Note that the referenced unique id is stored in library. Thus when calling the turbine via unique id, it should be prefixed by library name pywinda. See example below.

Parameters

- **uniqueID** – [*req*] Unique ID of the wind turbine as string
- **turbineType** – [*opt*] Type of turbine as string: 'SRT' or 'MRT'
- **diameter** – [*opt*] Diameter of the turbine as float
- **hubHeigt** – [*opt*] Hub height as a float
- **x_horizontal** – [*opt*] Horizontal coordinate of the turbine as float
- **y_vertical** – [*opt*] Vertical coordinate of the the turbine as float

Example

```
>>> from PyWinda import pywinda as pw
>>> curslack=pw.windfarm("uID_Curslack3")
>>> WT1=curslack.addTurbine('uID_WT14',turbineType='SRT',hubHeigt=120␣
↪)
>>> WT2=curslack.addTurbine('uID_WT15',turbineType='SRT',x_
↪horizontal=150,y_vertical=150)
>>> WT3=curslack.addTurbine('uID_WT16',turbineType='MRT',hubHeigt=200,
↪x_horizontal=300,y_vertical=300)
>>> WT3.diameter=150 #Assiging WT3 diameter after creation.
>>> print(WT3==pw.uID_WT16)
True
```

(continues on next page)

(continued from previous page)

```
>>> print(WT3.diameter)
150
>>> WT4=curslack.addTurbine('uID_WT16')
Traceback (most recent call last):
Exception: A wind turbine witht the same uniqe ID globally exists.␣
↪New turbine not added.
>>> WT5=curslack.addTurbine('uID WT16')
Traceback (most recent call last):
Exception: Name should be a string without spaces. The assignment␣
↪should be done via the UID and not the variable name.
```

---

property assets

Returns all the unique IDs of all the assets in the windfarm e.g. single rotor turbines, multirotor tubines, met masts, etc.

Parameters **None** –

Example

```
>>> from PyWinda import pywinda as pw
>>> curslack=pw.windfarm("uID_Curslack2")
>>> WT1=curslack.addTurbine('uID_WT11',turbineType='SRT',hubHeigt=120)
>>> WT2=curslack.addTurbine('uID_WT12',turbineType='SRT',hubHeigt=120)
>>> WT3=curslack.addTurbine('uID_MWT13',turbineType='MRT',
↪hubHeigt=200)
>>> print(curslack.assets)
['uID_WT11', 'uID_WT12', 'uID_MWT13']
```

---

assignEnvironment(*envName*)

Assigns an already created environment to the referenced wind farm. Parameters of the environment (e.g. temperature, pressure, wind regime etc.) can be assigned later. The environment would be callable via its unique name and the assigned variable by user. When using the unique Id, it should be prefixed witht he library name pywinda. See example.

Parameters **envName** – [*req*] unique nvironment name

Example

```
>>> from PyWinda import pywinda as pw
>>> DanTysk=pw.windfarm("DanTysk2")
>>> env=pw.environment('normal1')
>>> print(env.info.keys()) #shows some of the conditions of the␣
↪created environment
dict_keys(['Wind directions', 'Sectors', 'Wind speeds', 'Pressure',
↪'Temperature', 'Wind probability', 'Scale parameter of wind␣
↪distribution', 'Shape parameter of wind distribution'])
>>> print(env.info['Pressure'])
101325
>>> DanTysk.assignEnvironment('normal1')
```

(continues on next page)

(continued from previous page)

```
>>> DanTysk.assignEnvironment('normal2')
Traceback (most recent call last):
Exception: The wind farm [DanTysk2] already has assigned environment␣
↪[normal1]. New environment not added.
>>> print(pw.normal1==env)
True
```

---

coordinates(*assets=[]*)

Returns the data frame with all assets' x and y coordinates if the assets list is empty, otherwise only for the given set of assets.

Parameters **assets** – [*opt*] Unique ID or object name of the assets

Example

```
>>> from PyWinda import pywinda as pw
>>> Curslack = pw.windfarm("Curslack_farm01")
>>> WT1 = Curslack.addTurbine("C_WT11", x_horizontal=480331, y_
↪vertical=4925387)
>>> WT2 = Curslack.addTurbine("C_WT2", x_horizontal=480592, y_
↪vertical=4925253)
>>> WT3 = Curslack.addTurbine("C_WT3", x_horizontal=480886, y_
↪vertical=4925166)
>>> WT4 = Curslack.addTurbine("C_MWT4",x_horizontal=480573, y_
↪vertical=4925712)
>>> print(Curslack.coordinates())
Assets  x_coor   y_coor
C_WT11  480331  4925387
C_WT2   480592  4925253
C_WT3   480886  4925166
C_MWT4  480573  4925712
```

---

distances(*assets=[]*)

Returns the data frame with all the distances between assets in the wind farm or between those given in the assets list.

Parameters **assets** – [*opt*] Unique ID or object name of the assets

Example

```
>>> from PyWinda import pywinda as pw
>>> Curslack2 = pw.windfarm("Curslack_farm1")
>>> WT1 = Curslack2.addTurbine("C_WT01", x_horizontal=480331, y_
↪vertical=4925387)
>>> WT2 = Curslack2.addTurbine("C_WT02", x_horizontal=480592, y_
↪vertical=4925253)
>>> WT3 = Curslack2.addTurbine("C_WT03", x_horizontal=480886, y_
↪vertical=4925166)
>>> WT4 = Curslack2.addTurbine("C_MWT04",x_horizontal=480573, y_
↪vertical=4925712)
```

(continues on next page)

(continued from previous page)

```
>>> print(Curslack2.distances())
    Assets      C_WT01      C_WT02      C_WT03     C_MWT04
0   C_WT01    0.000000  293.388821  597.382624  405.202419
1   C_WT02  293.388821    0.000000  306.602348  459.393078
2   C_WT03  597.382624  306.602348    0.000000  629.352842
3  C_MWT04  405.202419  459.393078  629.352842    0.000000
```

---

property info

Returns a data frame containing all the information about the wind farm.

Parameters None –

Example

```
>>> from PyWinda import pywinda as pw
>>> curslack=pw.windfarm("uID_Curslack")
>>> WT1=curslack.addTurbine('uID_WT1',turbineType='SRT',hubHeigt=120,␣
↪x_horizontal=100,y_vertical=100)
>>> WT2=curslack.addTurbine('uID_WT2',turbineType='SRT',hubHeigt=120,␣
↪x_horizontal=150,y_vertical=150)
>>> WT3=curslack.addTurbine('uID_MWT3',turbineType='MRT',hubHeigt=200,
↪ x_horizontal=300,y_vertical=300)
>>> print(curslack.info)
         Property               Value
0       Unique ID        uID_Curslack
1    Created SRTs  [uID_WT1, uID_WT2]
2    Created MRTs           [uID_MWT3]
3  Number of SRTs                   2
4  Number of MRTs                   1
```

1.3 dfm_module module

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

p

pywinda, 3

INDEX