

```
[51] %matplotlib inline
```

# Data analysis with Python: maritime climate report

Dept. Mecánica de Estructuras e Ingeniería Hidráulica

E.T.S.I. Caminos, Canales y Puertos

Universidad de Granada

## Download files for the practice

[https://gdfa.ugr.es/share/pedro/curso\\_etsiccp.zip](https://gdfa.ugr.es/share/pedro/curso_etsiccp.zip)

## Block I - Read data

Open the file `SIMAR_tutorial.txt` and check the following items:

1. Has it header? How many lines?
2. How are the columns delimited (white space, tabulation, comma, etc)?
3. Are there null values?
4. Are there columns with dates?

## Activity I.1

Read the `SIMAR_tutorial.txt` file using the function **read\_csv** inside the **pandas** module.

```
[52] import pandas as pd

simar_tutorial = pd.read_csv('SIMAR_tutorial.txt')
print(simar_tutorial)
```

1							LATITUD	36.500
2							LONGITUD	-7.000
3							PROFUNDIDAD	Indefinida
4							LISTADO DE DATOS	
5	AA	MM	Hm0	Tm02	Tp	DirM	Hm0_V	...
6	1958	01	2.1	5.9	10.8	249.0	-9999	-...
7	1958	01	2.1	6.0	10.6	250.0	-9999	-...
8	1958	01	2.1	6.1	10.6	251.0	-9999	-...
9	1958	01	2.1	6.2	10.6	252.0	-9999	-...
10	1958	01	2.1	6.3	10.6	253.0	-9999	-...
11	1958	01	2.1	6.3	10.6	254.0	-9999	-...
12	1958	01	2.1	6.4	10.6	254.0	-9999	-...
13	1958	01	2.1	6.4	10.5	255.0	-9999	-...

Eliminar cabecera con el parámetro skiprows=n\_filas cabecera

```
[53] # Elimino cabecera
simar_tutorial = pd.read_csv('SIMAR_tutorial.txt',
                             skiprows=10)
print(simar_tutorial)
```

	AA	MM	Hm0	Tm02	Tp	DirM	Hm0_V	DirM_V	Hm0_F1	Tm02_F1
DirM_F1	Hm0_F2	Tm02_F2	DirM_F2	VelV	DirV					
0	1958	01	2.1	5.9	10.8	249.0	-9999	-...		...
1	1958	01	2.1	6.0	10.6	250.0	-9999	-...		...
2	1958	01	2.1	6.1	10.6	251.0	-9999	-...		...
3	1958	01	2.1	6.2	10.6	252.0	-9999	-...		...
4	1958	01	2.1	6.3	10.6	253.0	-9999	-...		...
5	1958	01	2.1	6.3	10.6	254.0	-9999	-...		...
6	1958	01	2.1	6.4	10.6	254.0	-9999	-...		...
7	1958	01	2.1	6.4	10.5	255.0	-9999	-...		...

Añadir delimitador de columnas con el parámetro delim\_whitespace=True

```
[54] # Añado delimitador
simar_tutorial = pd.read_csv('SIMAR_tutorial.txt',
                             skiprows=10,
                             delim_whitespace=True)
print(simar_tutorial)
```

	AA	MM	Hm0	Tm02	Tp	DirM	Hm0_V	DirM_V	Hm0_F1	Tm02_F1
DirM_F1	\									
0	1958	1	2.1	5.9	10.8	249.0	-9999	-9999	2.1	-9999
250.0										
1	1958	1	2.1	6.0	10.6	250.0	-9999	-9999	2.7	-9999
250.0										
2	1958	1	2.1	6.1	10.6	251.0	-9999	-9999	2.5	-9999
248.0										
3	1958	1	2.1	6.2	10.6	252.0	-9999	-9999	3.3	-9999
254.0										
4	1958	1	2.1	6.3	10.6	253.0	-9999	-9999	2.1	-9999

253.0										
5	1958	1	2.1	6.3	10.6	254.0	-9999	-9999	2.1	-9999
254.0										
6	1958	1	2.1	6.4	10.6	254.0	-9999	-9999	2.1	-9999
254.0										
7	1958	1	2.1	6.4	10.5	255.0	-9999	-9999	2.1	-9999
255.0										

	Hm0_F2	Tm02_F2	DirM_F2	VelV	DirV
0	-9999	-9999	-9999	7.9	172.0
1	-9999	-9999	-9999	7.5	170.0
2	-9999	-9999	-9999	7.1	169.0
3	-9999	-9999	-9999	6.7	167.0
4	-9999	-9999	-9999	6.3	162.0
5	-9999	-9999	-9999	6.4	157.0
6	-9999	-9999	-9999	6.4	151.0
7	-9999	-9999	-9999	6.6	141.0

Eliminar valores nulos con el parámetro na\_values=tipo de dato nulo

```
[55] # Elimino valores nulos
simar_tutorial = pd.read_csv('SIMAR_tutorial.txt',
                             skiprows=10,
                             delim_whitespace=True,
                             na_values=-9999)

print(simar_tutorial)
```

	AA	MM	Hm0	Tm02	Tp	DirM	Hm0_V	DirM_V	Hm0_F1	Tm02_F1
DirM_F1	\									
0	1958	1	2.1	5.9	10.8	249.0	NaN	NaN	2.1	NaN
250.0										
1	1958	1	2.1	6.0	10.6	250.0	NaN	NaN	2.7	NaN
250.0										
2	1958	1	2.1	6.1	10.6	251.0	NaN	NaN	2.5	NaN
248.0										
3	1958	1	2.1	6.2	10.6	252.0	NaN	NaN	3.3	NaN
254.0										
4	1958	1	2.1	6.3	10.6	253.0	NaN	NaN	2.1	NaN
253.0										
5	1958	1	2.1	6.3	10.6	254.0	NaN	NaN	2.1	NaN
254.0										
6	1958	1	2.1	6.4	10.6	254.0	NaN	NaN	2.1	NaN
254.0										
7	1958	1	2.1	6.4	10.5	255.0	NaN	NaN	2.1	NaN
255.0										

	Hm0_F2	Tm02_F2	DirM_F2	VelV	DirV
0	NaN	NaN	NaN	7.9	172.0
1	NaN	NaN	NaN	7.5	170.0
2	NaN	NaN	NaN	7.1	169.0
3	NaN	NaN	NaN	6.7	167.0
4	NaN	NaN	NaN	6.3	162.0
5	NaN	NaN	NaN	6.4	157.0

6	NaN	NaN	NaN	6.4	151.0
7	NaN	NaN	NaN	6.6	141.0

Leo las columnas de fecha como dato tipo fecha (DateTime)

```
[56] # Leo las columnas de fecha como dato tipo fecha (DateTime)
simar_tutorial = pd.read_csv('SIMAR_tutorial.txt',
                             skiprows=10,
                             delim_whitespace=True,
                             na_values=-9999,
                             parse_dates=[[0, 1]],
                             index_col=0)

print(simar_tutorial)
```

	Hm0	Tm02	Tp	DirM	Hm0_V	DirM_V	Hm0_F1	Tm02_F1	DirM_F1
\									
AA_MM									...
1958-01-01	2.1	5.9	10.8	249.0	NaN	NaN	2.1	NaN	250...
1958-01-01	2.1	6.0	10.6	250.0	NaN	NaN	2.7	NaN	250...
1958-01-01	2.1	6.1	10.6	251.0	NaN	NaN	2.5	NaN	248...
1958-01-01	2.1	6.2	10.6	252.0	NaN	NaN	3.3	NaN	254...
1958-01-01	2.1	6.3	10.6	253.0	NaN	NaN	2.1	NaN	253...
1958-01-01	2.1	6.3	10.6	254.0	NaN	NaN	2.1	NaN	254...
1958-01-01	2.1	6.4	10.6	254.0	NaN	NaN	2.1	NaN	254...
1958-01-01	2.1	6.4	10.5	255.0	NaN	NaN	2.1	NaN	255...

	Hm0_F2	Tm02_F2	DirM_F2	VelV	DirV
AA_MM					
1958-01-01	NaN	NaN	NaN	7.9	172.0
1958-01-01	NaN	NaN	NaN	7.5	170.0
1958-01-01	NaN	NaN	NaN	7.1	169.0
1958-01-01	NaN	NaN	NaN	6.7	167.0
1958-01-01	NaN	NaN	NaN	6.3	162.0
1958-01-01	NaN	NaN	NaN	6.4	157.0
1958-01-01	NaN	NaN	NaN	6.4	151.0
1958-01-01	NaN	NaN	NaN	6.6	141.0

## Exercise I.1

Read the SIMAR\_1052046.txt file

```
[114] simar = pd.read_table('SIMAR_1052046.txt',
                        skiprows=81,
                        delim_whitespace=True,
                        na_values=-99.9,
                        parse_dates=[[0, 1, 2, 3]],
                        index_col=0)

print(simar)
```

\		Hm0	Tm02	Tp	DirM	Hm0_V	DirM_V	Hm0_F1	Tm02_F1
AA_MM_DD_HH									...
1958-01-04	00:00:00	2.1	5.9	10.8	249.0	NaN	NaN	2.1	N...
1958-01-04	01:00:00	2.1	6.0	10.6	250.0	NaN	NaN	2.7	N...
1958-01-04	02:00:00	2.1	6.1	10.6	251.0	NaN	NaN	2.5	N...
1958-01-04	03:00:00	2.1	6.2	10.6	252.0	NaN	NaN	3.3	N...
1958-01-04	04:00:00	2.1	6.3	10.6	253.0	NaN	NaN	2.1	N...
1958-01-04	05:00:00	2.1	6.3	10.6	254.0	NaN	NaN	2.1	N...
1958-01-04	06:00:00	2.1	6.4	10.6	254.0	NaN	NaN	2.1	N...
1958-01-04	07:00:00	2.1	6.4	10.5	255.0	NaN	NaN	2.1	N...
1958-01-04	08:00:00	2.1	6.4	10.5	255.0	NaN	NaN	2.1	N...
1958-01-04	09:00:00	2.1	6.3	10.5	255.0	NaN	NaN	2.0	N...
1958-01-04	10:00:00	2.1	6.3	10.5	255.0	NaN	NaN	2.1	N...
1958-01-04	11:00:00	2.1	6.3	10.5	254.0	NaN	NaN	2.0	N...
1958-01-04	12:00:00	2.2	6.2	10.5	253.0	NaN	NaN	2.0	N...
1958-01-04	13:00:00	2.2	6.1	10.5	252.0	NaN	NaN	2.0	N...
1958-01-04	14:00:00	2.3	6.1	10.5	252.0	NaN	NaN	2.0	N...
1958-01-04	15:00:00	2.3	6.2	10.6	254.0	NaN	NaN	1.9	N...
1958-01-04	16:00:00	2.3	6.3	10.8	256.0	NaN	NaN	1.9	N...
1958-01-04	17:00:00	2.3	6.6	11.0	258.0	NaN	NaN	2.2	N...
1958-01-04	18:00:00	2.3	6.8	11.4	261.0	NaN	NaN	2.2	N...
1958-01-04	19:00:00	2.3	6.9	12.1	262.0	NaN	NaN	2.1	N...
1958-01-04	20:00:00	2.3	7.0	12.2	263.0	NaN	NaN	2.1	N...
1958-01-04	21:00:00	2.3	7.0	12.4	264.0	NaN	NaN	2.3	N...
1958-01-04	22:00:00	2.4	7.1	12.5	264.0	NaN	NaN	2.3	N...
1958-01-04	23:00:00	2.4	7.1	12.5	264.0	NaN	NaN	2.3	N...
1958-01-05	00:00:00	2.4	7.1	12.5	264.0	NaN	NaN	2.3	N...
1958-01-05	01:00:00	2.4	7.2	12.4	264.0	NaN	NaN	2.3	N...
1958-01-05	02:00:00	2.3	7.4	12.4	264.0	NaN	NaN	2.3	N...
1958-01-05	03:00:00	2.3	7.6	12.4	265.0	NaN	NaN	2.2	N...
1958-01-05	04:00:00	2.3	7.8	12.4	265.0	NaN	NaN	2.2	N...
1958-01-05	05:00:00	2.2	7.8	12.2	265.0	NaN	NaN	2.2	N...

## Activity 1.2

Represent the location of a point in a map

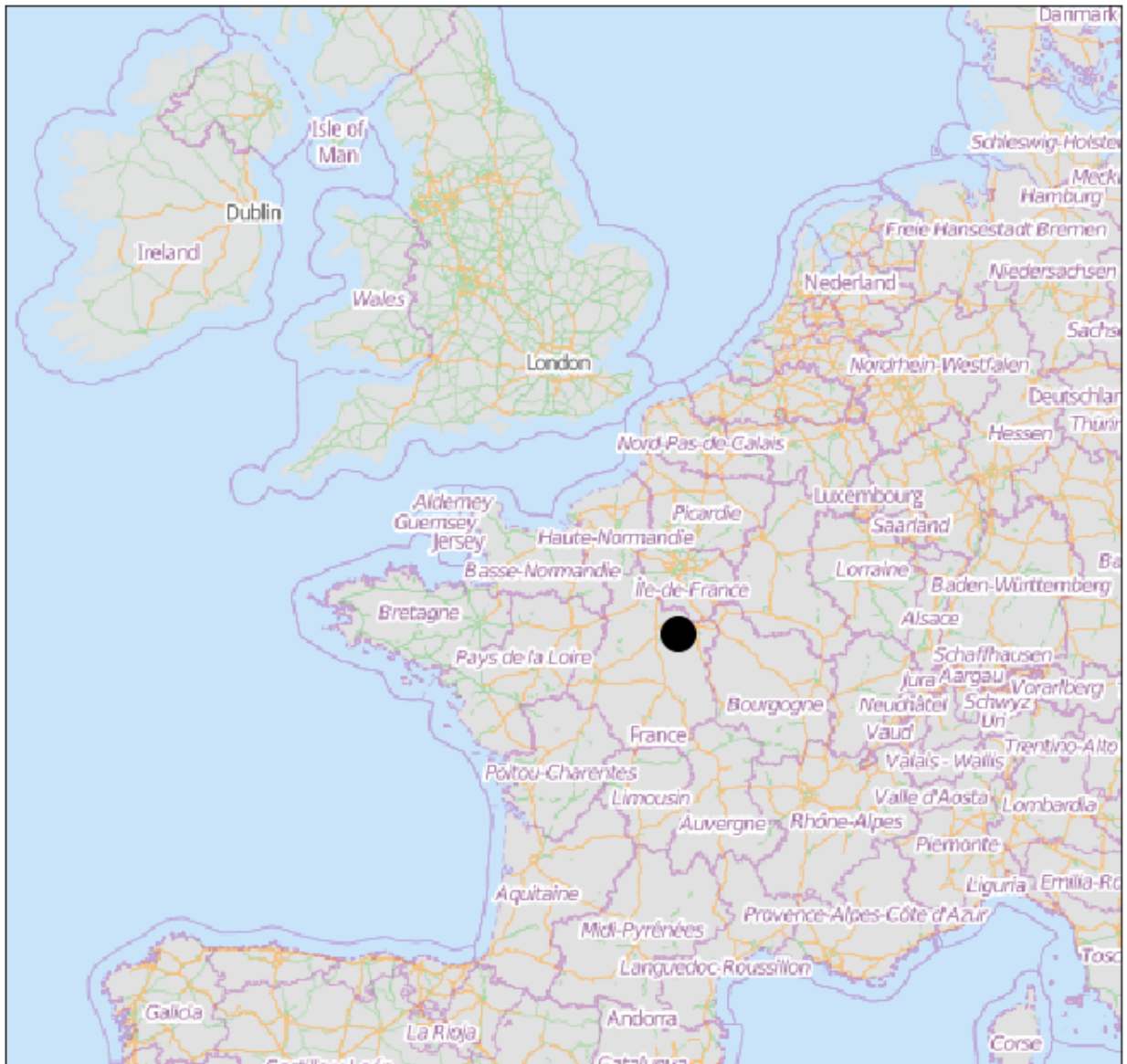
```
[58] import smopy
import matplotlib.pyplot as plt

### Bloque II Mapa
# Create de figure
figure = plt.figure(figsize=(16, 12))
ax = plt.axes()
plt.xticks([])
plt.yticks([])

#Plot the map and the point
map = smopy.Map((45., 0., 50., 5.), z=5)
x, y = map.to_pixels(48, 2.3)
map.show_mpl(ax)
ax.plot(x, y, 'ok', ms=20)
```

```
# Show figure
plt.show()

#Save figure (optional)
figure.savefig('point_location.png')
```



## Exercise I.2

Represent the location of the SIMAR\_1052046 data

```
[59] %matplotlib inline
import smopy
import matplotlib.pyplot as plt

figure = plt.figure(figsize=(16, 12))
ax = plt.axes()
plt.xticks([])
plt.yticks([])
```

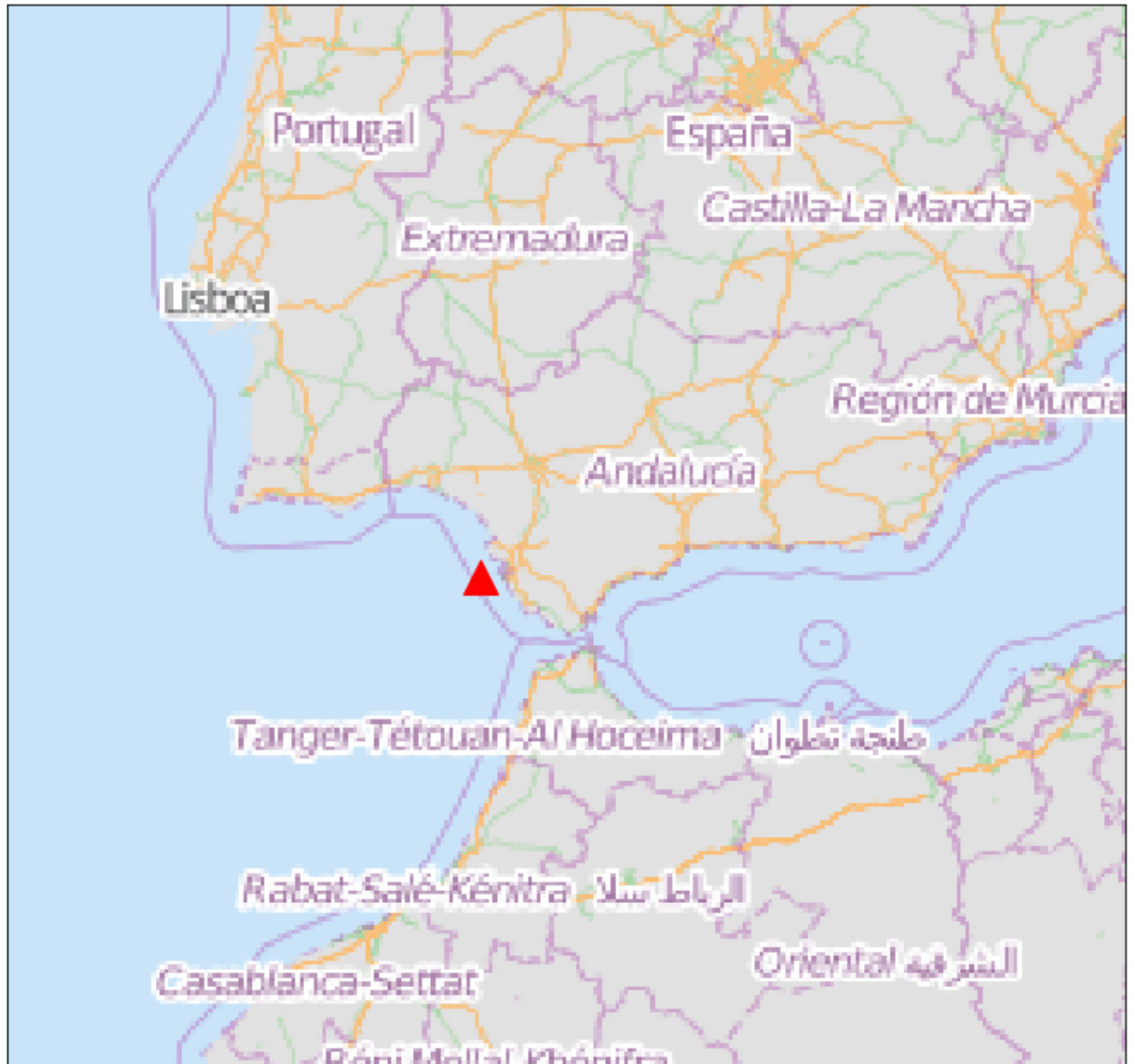
```

map = smopy.Map((36., -7., 37., -5.), z=5)
x, y = map.to_pixels(36.5, -6.5)
map.show_mpl(ax)

ax.plot(x, y, '^r', ms=20)

figure.savefig('simar_cadiz_location.png')

```



## Block II - Quick summary of the data

### Activity II.1

What is the maximum registered significant wave height (Hm0)?

```

[96] Hm0_max = np.max(simar.loc[:, 'Hm0'])
      print(Hm0_max)

```

What is the more frequent direction of origin of the wind (DirV)?

```
[61] from scipy import stats

wind_dir = stats.mode(simar.loc[:, 'DirV'])
print(wind_dir)

ModeResult(mode=array([270.]), count=array([4076]))
```

## Exercise II.1

Calculate the following values of the SIMAR data set:

- **EASY** The maximum registered value of Wind Velocity (VelV)
- **EASY** The average direction of origin of the wave (DirM) using the function **np.mean()**
- **MEDIUM** The minum registered peak period (Tp) between 2001 and 2005
- **HARD** The instants where the significant wave height (Hm0) is higher than 3 meters in 2017. (Hint: Use the function **np.where()**)

1. The maximum registered value of Wind Velocity (VelV)

```
[62] max_wind_vel = np.max(simar.loc[:, 'VelV'])
print(max_wind_vel)
```

25.0

2. The average direction of origin of the wave (DirM)

```
[63] average_dirM = np.mean(simar.loc[:, 'DirM'])
print(average_dirM)
```

242.829821335

3. The minum registered peak period (Tp) between 2001 and 2005

```
[64] min_tp = np.min(simar.loc['2001':'2005', 'Tp'])
print(min_tp)
```

2.4



#### 4. The instants where the significant wave height (Hm0) is higher than 3 meters in 2017

```
[65] # Crop the simar
simar_crop = simar.loc['2017', 'Hm0']
print(simar_crop)
```

```
AA_MM_DD_HH
2017-01-01 00:00:00    1.4
2017-01-01 01:00:00    1.4
2017-01-01 02:00:00    1.4
2017-01-01 03:00:00    1.4
2017-01-01 04:00:00    1.3
2017-01-01 05:00:00    1.4
2017-01-01 06:00:00    1.4
2017-01-01 07:00:00    1.4
2017-01-01 08:00:00    1.3
2017-01-01 09:00:00    1.3
2017-01-01 10:00:00    1.2
2017-01-01 11:00:00    1.2
2017-01-01 12:00:00    1.1
2017-01-01 13:00:00    1.1
2017-01-01 14:00:00    1.1
2017-01-01 15:00:00    1.1
2017-01-01 16:00:00    1.1
2017-01-01 17:00:00    1.1
2017-01-01 18:00:00    1.0
2017-01-01 19:00:00    1.0
2017-01-01 20:00:00    0.9
2017-01-01 21:00:00    0.9
2017-01-01 22:00:00    1.0
2017-01-01 23:00:00    1.1
2017-01-02 00:00:00    1.2
2017-01-02 01:00:00    1.3
2017-01-02 02:00:00    1.4
2017-01-02 03:00:00    1.5
2017-01-02 04:00:00    1.5
2017-01-02 05:00:00    1.5
...
2017-08-05 19:00:00    0.9
```

```
[66] # Obtain the positions that match the condition
[pos, ] = np.where(simar_crop > 5)
print(pos)
```

```
[2591 2592 2593 2594 2595 2596 2597 2598 2599 2600 2601 2602 2603 2604
 2612 2613 2614 2615 2616 2617 2618 2619 2620 3340 3341 3342 3343 3344
 3345]
```

```
[67] # Evaluate the positions
instants = simar_crop[pos]
print(instants)
```

```

AA_MM_DD_HH
2017-04-19 23:00:00    5.1
2017-04-20 00:00:00    5.2
2017-04-20 01:00:00    5.4
2017-04-20 02:00:00    5.6
2017-04-20 03:00:00    5.9
2017-04-20 04:00:00    6.0
2017-04-20 05:00:00    6.0
2017-04-20 06:00:00    5.9
2017-04-20 07:00:00    5.9
2017-04-20 08:00:00    5.9
2017-04-20 09:00:00    5.8
2017-04-20 10:00:00    5.6
2017-04-20 11:00:00    5.4
2017-04-20 12:00:00    5.2
2017-04-20 20:00:00    5.2
2017-04-20 21:00:00    5.4
2017-04-20 22:00:00    5.6
2017-04-20 23:00:00    5.8
2017-04-21 00:00:00    5.9
2017-04-21 01:00:00    5.9
2017-04-21 02:00:00    5.9
2017-04-21 03:00:00    5.7
2017-04-21 04:00:00    5.3
2017-05-21 04:00:00    5.3
2017-05-21 05:00:00    5.4
2017-05-21 06:00:00    5.2
2017-05-21 07:00:00    5.2
2017-05-21 08:00:00    5.2
2017-05-21 09:00:00    5.1
Name: Hm0, dtype: float64

```

## Activity II.2

Describe function of pandas

```

[68] table = simar.describe()
      print(table)

```

	Hm0	Tm02	Tp	DirM \
count	520976.000000	520976.000000	520976.000000	520976.000000
mean	1.036288	4.870426	8.109552	242.829821
std	0.708980	1.654451	3.095294	67.704083
min	0.000000	1.900000	0.000000	0.000000
25%	0.600000	3.700000	5.200000	228.000000
50%	0.900000	4.400000	8.100000	275.000000
75%	1.300000	5.700000	10.400000	283.000000
max	8.200000	14.200000	20.900000	360.000000

	Hm0_V	DirM_V	Hm0_F1	Tm02_F1 \
count	205448.000000	205448.000000	454655.000000	95677.000000
mean	1.164350	220.502478	0.693246	7.142502
std	0.905604	95.483967	0.624139	2.637794

min	0.000000	0.000000	0.000000	0.000000
25%	0.600000	126.000000	0.300000	4.800000
50%	0.900000	258.000000	0.500000	6.800000
75%	1.500000	299.000000	0.900000	9.000000
max	10.700000	360.000000	9.100000	23.900000

	DirM_F1	Hm0_F2	Tm02_F2	DirM_F2 \
count	454655.000000	168031.000000	66633.000000	168031.000000
mean	266.705678	0.336786	7.832900	256.960710
std	46.092148	0.255770	3.558114	62.740074
min	0.000000	0.000000	0.000000	0.000000
25%	271.000000	0.200000	4.600000	250.000000
50%	280.000000	0.300000	7.700000	281.000000
75%	286.000000	0.400000	10.100000	290.000000
max	360.000000	3.700000	23.900000	360.000000

	VelV	DirV
count	520976.000000	520976.000000

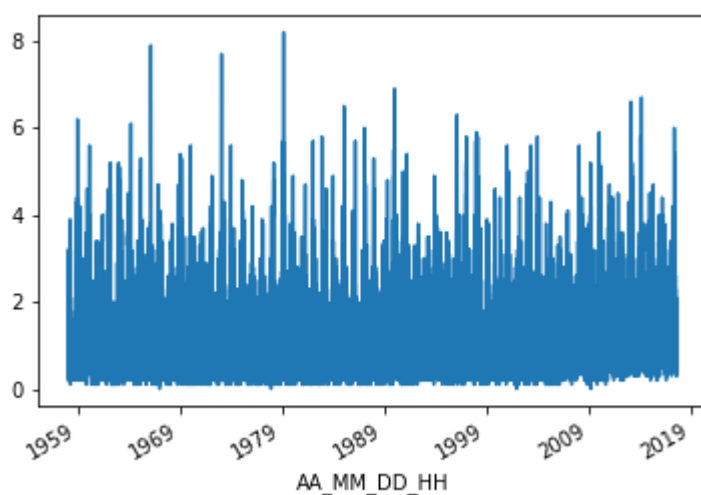
## Block III - Plotting data for analysis

### Activity III.1

Plot time series of Hm0 using pandas (**quick** but **not customizable** graphics)

```
[69] simar.loc[:, 'Hm0'].plot()
```

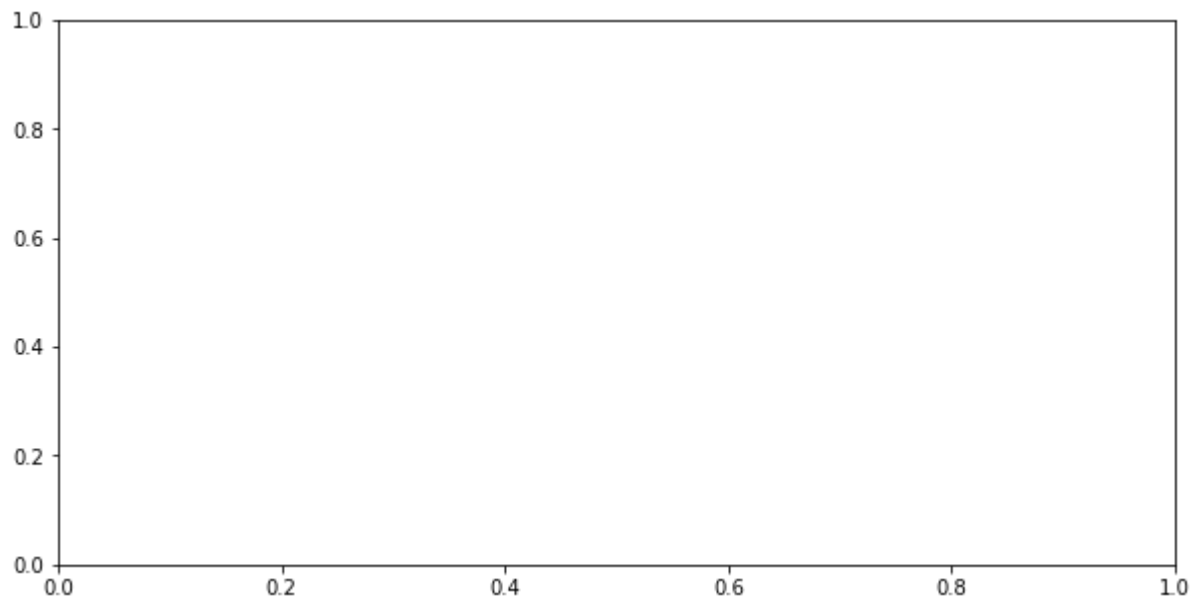
```
<matplotlib.axes._subplots.AxesSubplot at 0x12a00898>
```



### Activity III.2

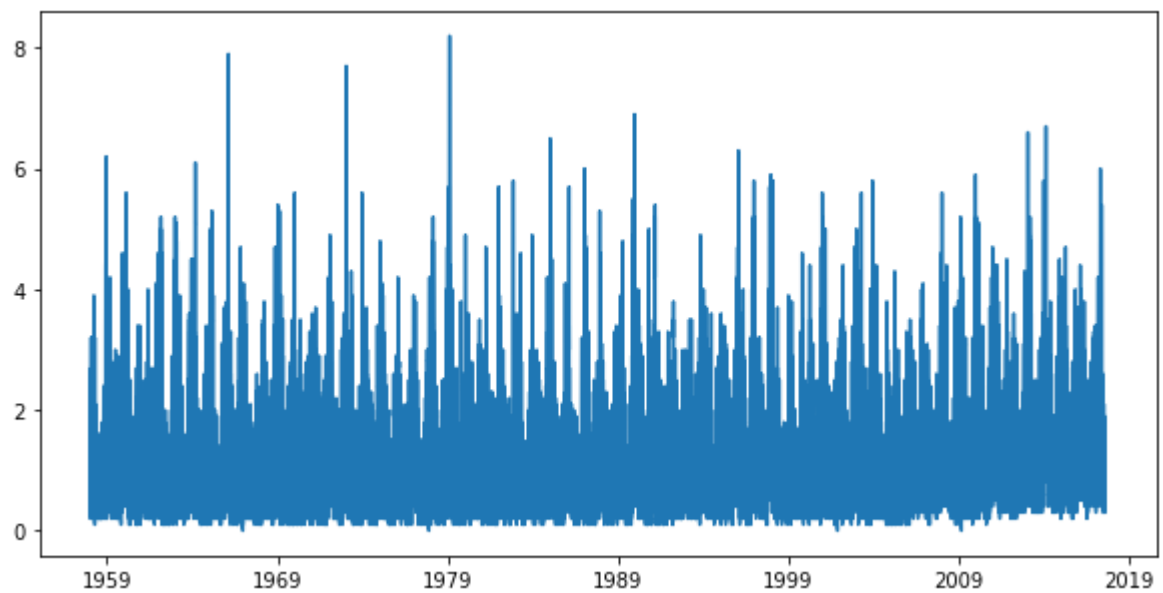
Plotting time series using matplotlib

```
[97] fig = plt.figure(figsize=(10, 5))
      ax = plt.axes()
      plt.show()
```



```
[98] fig = plt.figure(figsize=(10, 5))
      ax = plt.axes()

      ax.plot(simar.loc[:, 'Hm0'])
      plt.show()
```

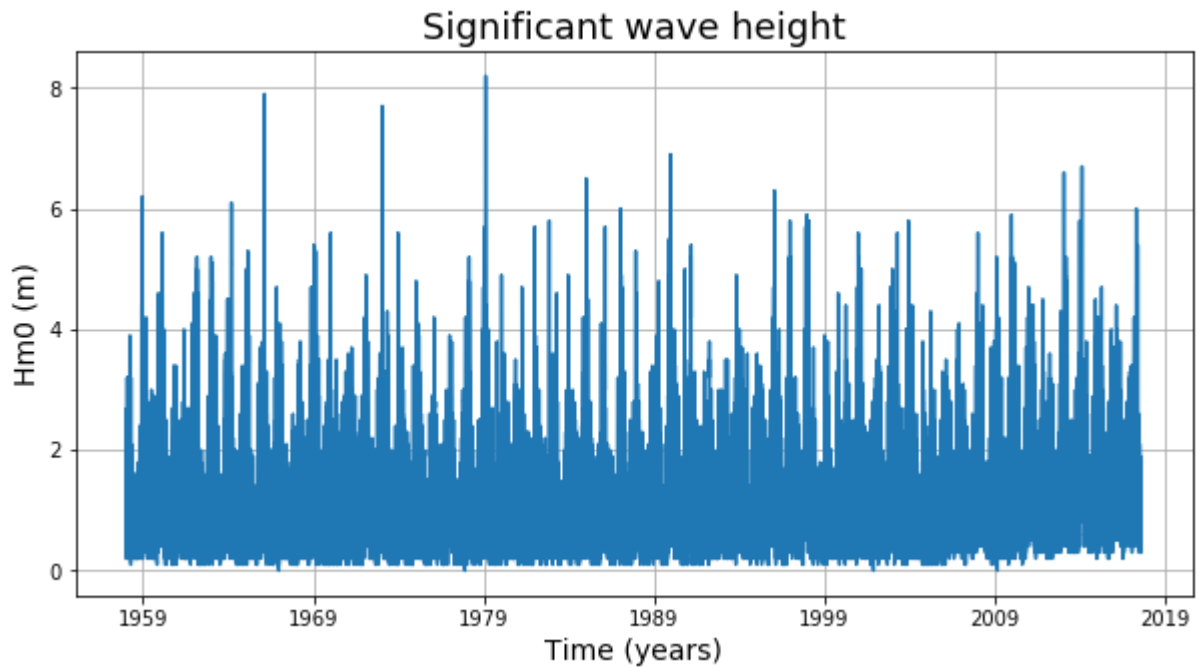


```
[99] fig5 = plt.figure(figsize=(10, 5))
      ax = plt.axes()

      ax.plot(simar.loc[:, 'Hm0'])

      ax.set_xlabel('Time (years)', fontsize=14)
      ax.set_ylabel('Hm0 (m)', fontsize=14)
```

```
plt.title('Significant wave height', fontsize=18)
ax.grid()
plt.show()
fig5.savefig('Hm0_plot.jpg')
```

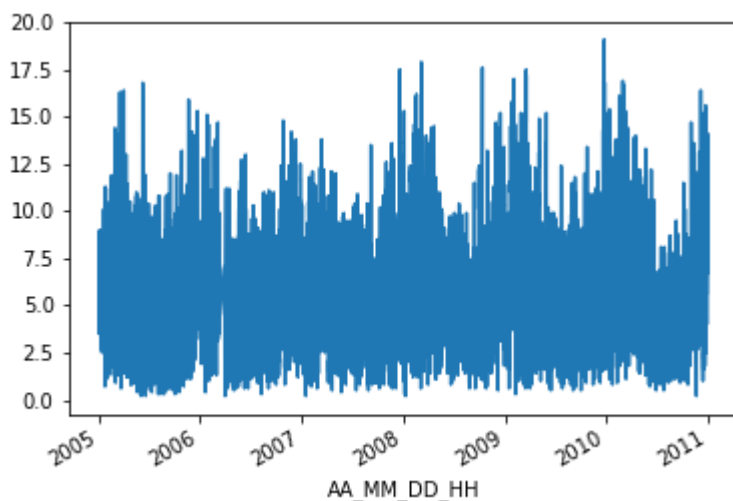


## Exercise III.2

Plot time series of another variable between 2005 and 2010

```
[71] simar.loc['2005':'2010', 'VelV'].plot()
```

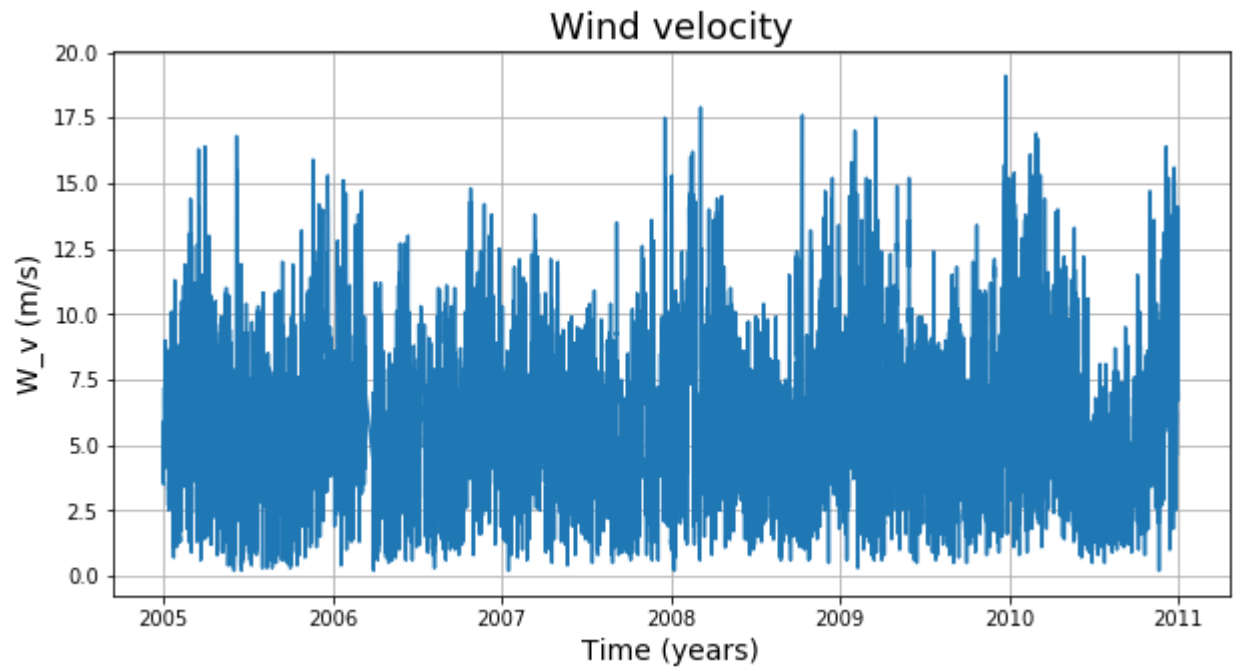
<matplotlib.axes.\_subplots.AxesSubplot at 0x1201a1d0>



```
[72] fig = plt.figure(figsize=(10, 5))
      ax = plt.axes()

      ax.plot(simar.loc['2005':'2010', 'VelV'])
```

```
ax.set_xlabel('Time (years)', fontsize=14)
ax.set_ylabel('W_v (m/s)', fontsize=14)
plt.title('Wind velocity', fontsize=18)
ax.grid()
```

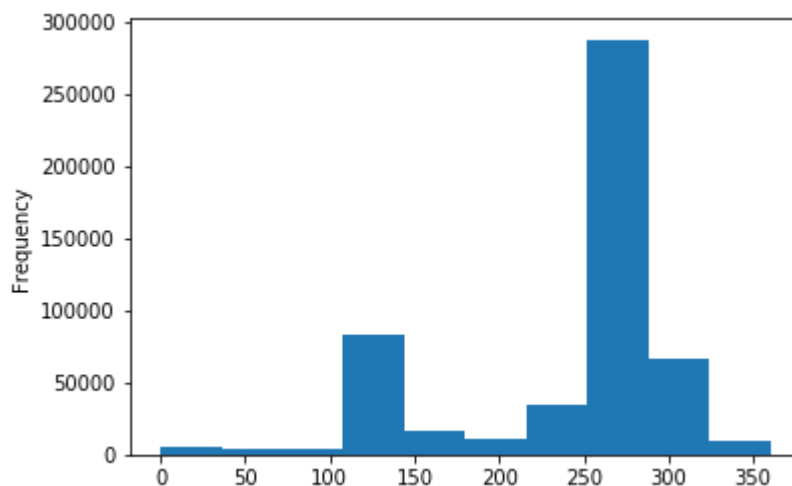


## Activity III.3

Plot histogram of DirM using Pandas

```
[73] simar.loc[:, 'DirM'].plot.hist(bins=10)
```

<matplotlib.axes.\_subplots.AxesSubplot at 0x21e94198>



## Exercise III.4



2015-01-01 00:00:00	0.7	3.8	10.8	338.0	0.4	38.0	0.4	10...
2015-01-01 01:00:00	0.7	3.7	10.8	346.0	0.5	37.0	0.5	10...
2015-01-01 02:00:00	0.8	3.7	10.7	354.0	0.5	38.0	0.5	10...
2015-01-01 03:00:00	0.8	3.7	10.6	0.0	0.5	38.0	0.5	10...
2015-01-01 04:00:00	0.9	3.7	10.5	8.0	0.6	41.0	0.5	10...
2015-01-01 05:00:00	0.9	3.7	10.4	18.0	0.7	41.0	0.5	9...
2015-01-01 06:00:00	1.0	3.6	10.3	24.0	0.8	44.0	0.5	9...
2015-01-01 07:00:00	1.1	3.7	10.2	26.0	0.9	43.0	0.5	9...
2015-01-01 08:00:00	1.1	3.7	4.3	29.0	1.0	43.0	0.5	9...
2015-01-01 09:00:00	1.2	3.7	4.4	32.0	1.1	43.0	0.5	9...
2015-01-01 10:00:00	1.3	3.8	4.5	35.0	1.2	44.0	0.5	9...
2015-01-01 11:00:00	1.4	3.8	4.7	38.0	1.3	45.0	0.5	9...
2015-01-01 12:00:00	1.5	3.8	4.8	40.0	1.3	46.0	0.5	9...
2015-01-01 13:00:00	1.4	3.8	4.8	41.0	1.2	49.0	0.5	9...
2015-01-01 14:00:00	1.2	3.8	4.8	40.0	NaN	NaN	1.0	4...
2015-01-01 15:00:00	1.1	3.7	9.8	36.0	0.3	124.0	0.8	4...
2015-01-01 16:00:00	1.0	3.7	9.8	29.0	0.3	119.0	0.7	3...
2015-01-01 17:00:00	0.9	3.7	9.7	45.0	0.5	114.0	0.6	9...
2015-01-01 18:00:00	1.0	3.8	9.6	130.0	0.7	90.0	0.6	9...
2015-01-01 19:00:00	1.1	3.9	9.6	145.0	0.9	118.0	0.6	9...
2015-01-01 20:00:00	1.2	4.1	9.5	140.0	1.0	123.0	0.6	9...
2015-01-01 21:00:00	1.3	4.1	5.4	135.0	1.1	122.0	0.6	9...
2015-01-01 22:00:00	1.4	4.2	5.7	131.0	1.2	121.0	0.6	9...
2015-01-01 23:00:00	1.4	4.2	5.7	129.0	1.2	120.0	0.6	9...
2015-01-02 00:00:00	1.4	4.2	5.8	127.0	1.3	119.0	0.6	9...
2015-01-02 01:00:00	1.4	4.2	5.8	125.0	1.3	118.0	0.6	8...
2015-01-02 02:00:00	1.4	4.2	5.8	123.0	1.3	116.0	0.6	8...
2015-01-02 03:00:00	1.4	4.2	5.7	122.0	1.3	115.0	0.6	8...
2015-01-02 04:00:00	1.4	4.1	5.6	122.0	1.2	115.0	0.5	8...

```
[77] # Extract Hm0
      simar_2015_01_Hm0 = simar_2015_01.loc[:, 'Hm0']
      print(simar_2015_01_Hm0)
```

AA_MM_DD_HH	
2015-01-01 00:00:00	0.7
2015-01-01 01:00:00	0.7
2015-01-01 02:00:00	0.8
2015-01-01 03:00:00	0.8
2015-01-01 04:00:00	0.9
2015-01-01 05:00:00	0.9
2015-01-01 06:00:00	1.0
2015-01-01 07:00:00	1.1
2015-01-01 08:00:00	1.1
2015-01-01 09:00:00	1.2
2015-01-01 10:00:00	1.3
2015-01-01 11:00:00	1.4
2015-01-01 12:00:00	1.5
2015-01-01 13:00:00	1.4
2015-01-01 14:00:00	1.2
2015-01-01 15:00:00	1.1
2015-01-01 16:00:00	1.0
2015-01-01 17:00:00	0.9
2015-01-01 18:00:00	1.0
2015-01-01 19:00:00	1.1
2015-01-01 20:00:00	1.2
2015-01-01 21:00:00	1.3



```

2015-01-01 22:00:00    1.4
2015-01-01 23:00:00    1.4
2015-01-02 00:00:00    1.4
2015-01-02 01:00:00    1.4
2015-01-02 02:00:00    1.4
2015-01-02 03:00:00    1.4
2015-01-02 04:00:00    1.4
2015-01-02 05:00:00    1.4
...

```

```

[104] # Repeat for the the month of august
      indices = np.where(simar.index.year == 2015)
      simar_2015 = simar.iloc[indices[0], :]
      indices = np.where(simar_2015.index.month == 8)
      simar_2015_8 = simar_2015.iloc[indices[0],:]
      simar_2015_8_Hm0 = simar_2015_8.loc[:, 'Hm0']
      print(simar_2015_8_Hm0)

```

```

AA_MM_DD_HH
2015-08-01 00:00:00    1.5
2015-08-01 01:00:00    1.4
2015-08-01 02:00:00    1.3
2015-08-01 03:00:00    1.2
2015-08-01 04:00:00    1.1
2015-08-01 05:00:00    1.1
2015-08-01 06:00:00    1.0
2015-08-01 07:00:00    1.0
2015-08-01 08:00:00    0.9
2015-08-01 09:00:00    0.9
2015-08-01 10:00:00    0.9
2015-08-01 11:00:00    0.9
2015-08-01 12:00:00    0.8
2015-08-01 13:00:00    0.7
2015-08-01 14:00:00    0.7
2015-08-01 15:00:00    0.7
2015-08-01 16:00:00    0.7
2015-08-01 17:00:00    0.7
2015-08-01 18:00:00    0.8
2015-08-01 19:00:00    0.8
2015-08-01 20:00:00    0.7
2015-08-01 21:00:00    0.7
2015-08-01 22:00:00    0.7
2015-08-01 23:00:00    0.7
2015-08-02 00:00:00    0.7
2015-08-02 01:00:00    0.6
2015-08-02 02:00:00    0.6
2015-08-02 03:00:00    0.6
2015-08-02 04:00:00    0.6
2015-08-02 05:00:00    0.6
...
2015-08-30 18:00:00    1.9

```

```

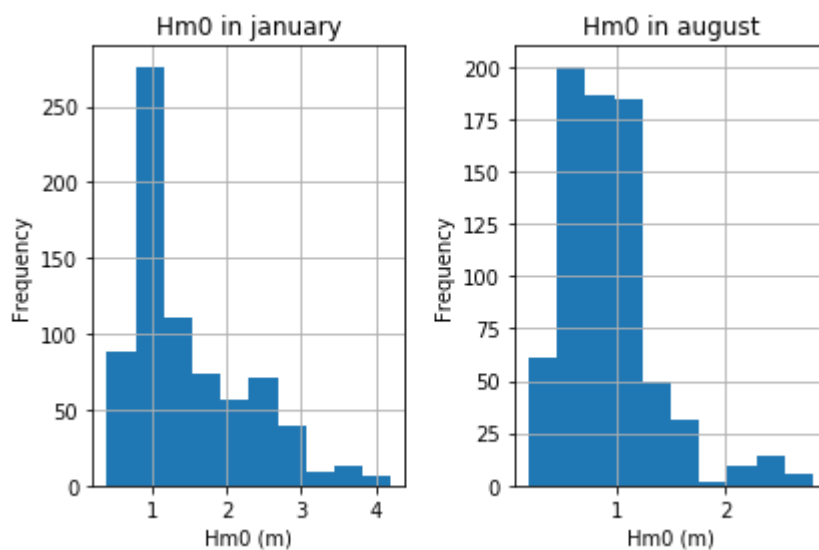
[79] # An easier and faster way
      simar_2015_1_Hm0 = simar['2015-01']['Hm0']
      simar_2015_8_Hm0 = simar['2015-08']['Hm0']

```

```
[80] # Plot subplot figure
plt.subplot(1, 2, 1)
simar_2015_1_Hm0.plot.hist(bins=10)
plt.xlabel('Hm0 (m)')
plt.ylabel('Frequency')
plt.title('Hm0 in january')
plt.grid(True)

plt.subplot(1, 2, 2)
simar_2015_8_Hm0.plot.hist(bins=10)
plt.xlabel('Hm0 (m)')
plt.ylabel('Frequency')
plt.title('Hm0 in august')
plt.grid(True)

plt.tight_layout()
plt.show()
```

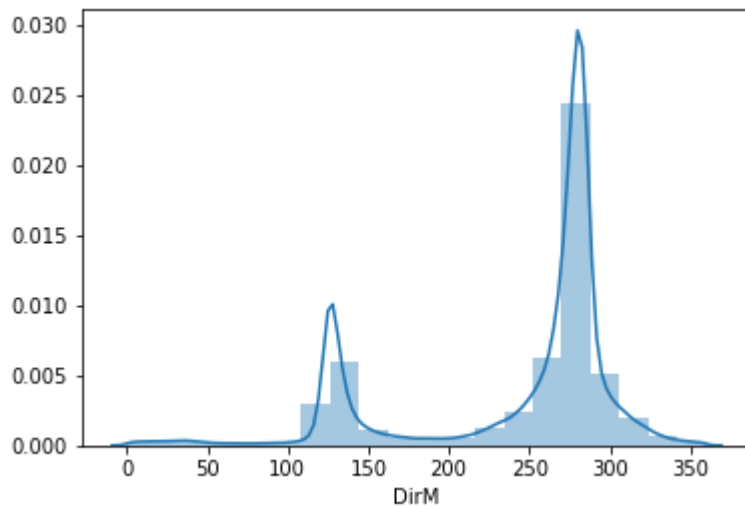


## Activity III.4

Plot histogram of DirM using Seaborn

```
[81] import seaborn as sns

values = simar.loc[:, 'DirM']
ax = sns.distplot(values, bins = 20)
```

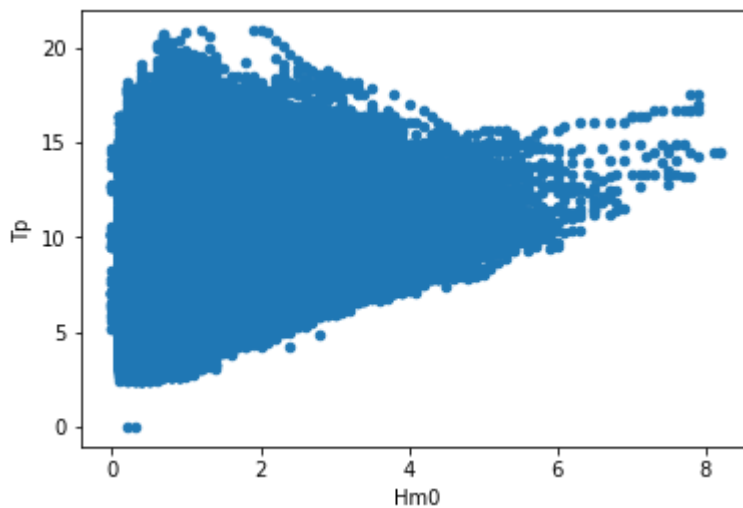


## Activity III.5

Dispersion graphic using Pandas between Hm0 and Tp

```
[82] simar.plot.scatter('Hm0', 'Tp')
```

<matplotlib.axes.\_subplots.AxesSubplot at 0x1dc11160>



## Exercise III.5

Compare and plot the dispersion graphic between VelV and DirV 1959 and 2016

```
[119] # Extraigo el año
indices = np.where(simar.index.year == 1959)
simar_1959 = simar.iloc[indices[0], :]
print(simar_1959)
```

\		Hm0	Tm02	Tp	DirM	Hm0_V	DirM_V	Hm0_F1	Tm02_F1
AA_MM_DD_HH									...
1959-01-01 00:00:00	0.6	6.2	9.9	276.0	NaN	NaN	0.6	N...	
1959-01-01 01:00:00	0.6	6.7	9.9	278.0	NaN	NaN	0.6	N...	
1959-01-01 02:00:00	0.6	7.2	9.8	279.0	NaN	NaN	0.6	N...	
1959-01-01 03:00:00	0.6	7.5	9.6	280.0	NaN	NaN	0.6	N...	
1959-01-01 04:00:00	0.6	7.7	9.5	280.0	NaN	NaN	0.6	N...	
1959-01-01 05:00:00	0.6	7.7	9.5	280.0	NaN	NaN	0.6	N...	
1959-01-01 06:00:00	0.6	7.8	9.4	280.0	NaN	NaN	0.6	N...	
1959-01-01 07:00:00	0.6	7.9	9.4	280.0	NaN	NaN	0.6	N...	
1959-01-01 08:00:00	0.6	7.9	9.4	280.0	NaN	NaN	0.6	N...	
1959-01-01 09:00:00	0.6	7.9	9.4	279.0	NaN	NaN	0.6	N...	
1959-01-01 10:00:00	0.6	7.9	9.4	279.0	NaN	NaN	0.6	N...	
1959-01-01 11:00:00	0.6	8.0	9.4	280.0	NaN	NaN	0.6	N...	
1959-01-01 12:00:00	0.6	8.3	9.3	280.0	NaN	NaN	0.6	N...	
1959-01-01 13:00:00	0.6	8.5	9.3	281.0	NaN	NaN	0.6	N...	
1959-01-01 14:00:00	0.6	8.6	9.3	281.0	NaN	NaN	0.6	N...	
1959-01-01 15:00:00	0.6	8.5	9.3	281.0	NaN	NaN	0.6	N...	
1959-01-01 16:00:00	0.6	8.5	9.3	281.0	NaN	NaN	0.6	N...	
1959-01-01 17:00:00	0.6	8.4	9.3	281.0	NaN	NaN	0.6	N...	
1959-01-01 18:00:00	0.5	8.3	9.3	280.0	NaN	NaN	0.5	N...	
1959-01-01 19:00:00	0.5	8.2	9.3	280.0	NaN	NaN	0.5	N...	
1959-01-01 20:00:00	0.5	8.1	9.3	280.0	NaN	NaN	0.5	N...	
1959-01-01 21:00:00	0.5	8.0	9.3	280.0	NaN	NaN	0.5	N...	
1959-01-01 22:00:00	0.5	7.9	9.3	280.0	NaN	NaN	0.5	N...	
1959-01-01 23:00:00	0.5	7.8	9.4	280.0	NaN	NaN	0.5	N...	
1959-01-02 00:00:00	0.5	7.8	9.4	280.0	NaN	NaN	0.5	N...	
1959-01-02 01:00:00	0.5	7.7	9.4	280.0	NaN	NaN	0.5	N...	
1959-01-02 02:00:00	0.5	7.6	9.5	280.0	NaN	NaN	0.5	N...	
1959-01-02 03:00:00	0.5	7.6	9.5	280.0	NaN	NaN	0.5	N...	
1959-01-02 04:00:00	0.5	7.6	9.6	281.0	NaN	NaN	0.4	N...	
1959-01-02 05:00:00	0.5	7.5	9.6	281.0	NaN	NaN	0.3	N...	

```
[120] # Extraigo el descriptor VelV
      simar_1959_velv = simar_1959.loc[:, 'VelV']
      print(simar_1959_velv)
```

AA_MM_DD_HH	
1959-01-01 00:00:00	3.1
1959-01-01 01:00:00	3.4
1959-01-01 02:00:00	3.6
1959-01-01 03:00:00	3.5
1959-01-01 04:00:00	3.6
1959-01-01 05:00:00	3.5
1959-01-01 06:00:00	3.3
1959-01-01 07:00:00	3.0
1959-01-01 08:00:00	2.9
1959-01-01 09:00:00	2.9
1959-01-01 10:00:00	3.0
1959-01-01 11:00:00	3.2
1959-01-01 12:00:00	3.3
1959-01-01 13:00:00	3.1
1959-01-01 14:00:00	3.0
1959-01-01 15:00:00	2.7
1959-01-01 16:00:00	2.6
1959-01-01 17:00:00	2.5

```

1959-01-01 18:00:00    2.4
1959-01-01 19:00:00    2.1
1959-01-01 20:00:00    2.2
1959-01-01 21:00:00    2.7
1959-01-01 22:00:00    3.1
1959-01-01 23:00:00    3.3
1959-01-02 00:00:00    3.1
1959-01-02 01:00:00    3.1
1959-01-02 02:00:00    3.2
1959-01-02 03:00:00    3.6
1959-01-02 04:00:00    3.8
1959-01-02 05:00:00    4.0
...

```

```

[83] Velv_1959 = simar.loc['1959', 'VelV']
     Dirv_1959 = simar.loc['1959', 'DirV']
     Velv_2016 = simar.loc['2016', 'VelV']
     Dirv_2016 = simar.loc['2016', 'DirV']

```

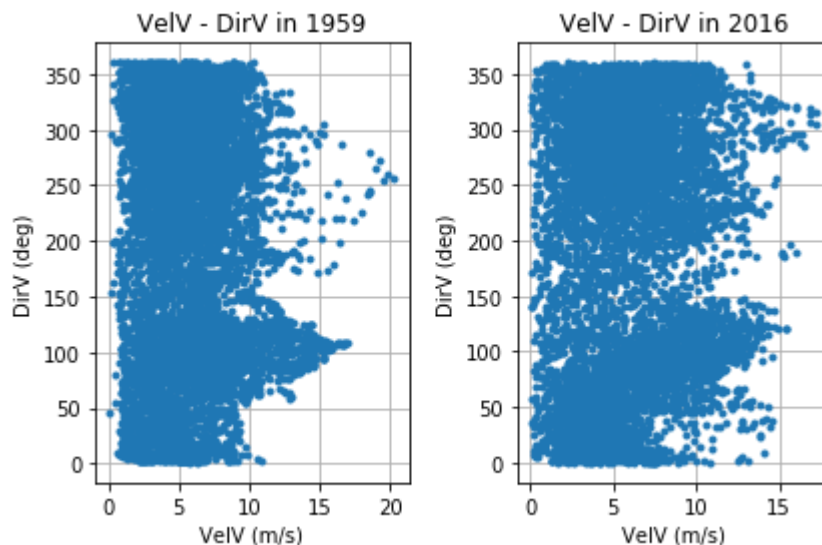
```

[84] # Plot subplot figure
plt.subplot(1, 2, 1)
plt.plot(Velv_1959, Dirv_1959, '.')
plt.xlabel('VelV (m/s)')
plt.ylabel('DirV (deg)')
plt.title('VelV - DirV in 1959')
plt.grid(True)

plt.subplot(1, 2, 2)
plt.plot(Velv_2016, Dirv_2016, '.')
plt.xlabel('VelV (m/s)')
plt.ylabel('DirV (deg)')
plt.title('VelV - DirV in 2016')
plt.grid(True)

plt.tight_layout()
plt.show()

```

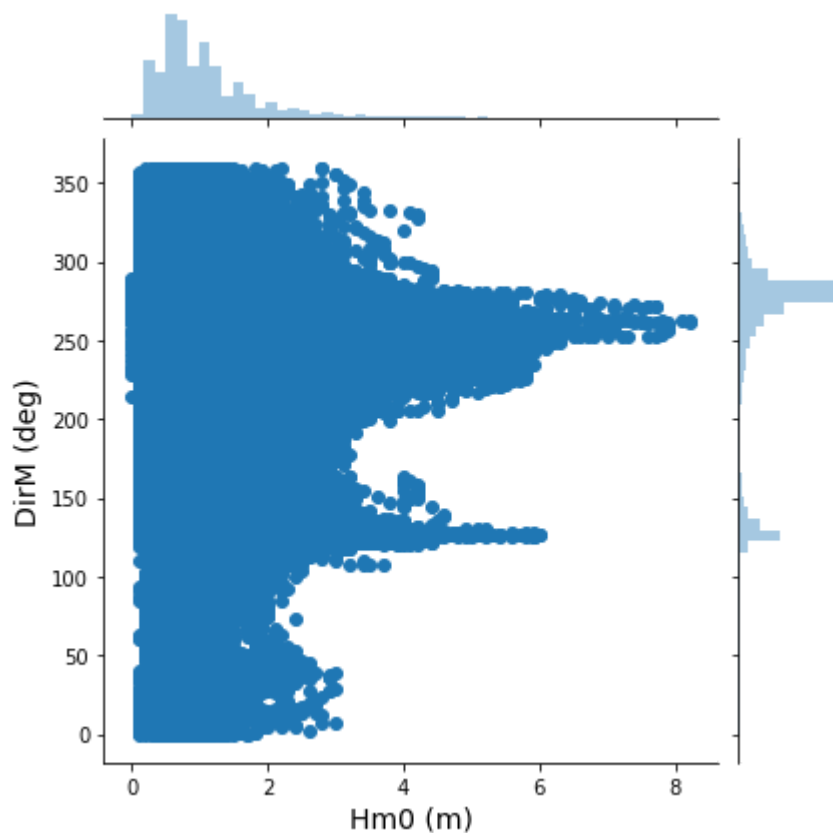


## Activity III.5

Dispersion graphic using Seaborn between Hm0 and DirM

```
[85] ax = sns.jointplot('Hm0', 'DirM', simar)
      ax.ax_joint.set_xlabel('Hm0 (m)', fontsize=14)
      ax.ax_joint.set_ylabel('DirM (deg)', fontsize=14)
```

```
Text(27.125,0.5,u'DirM (deg)')
```



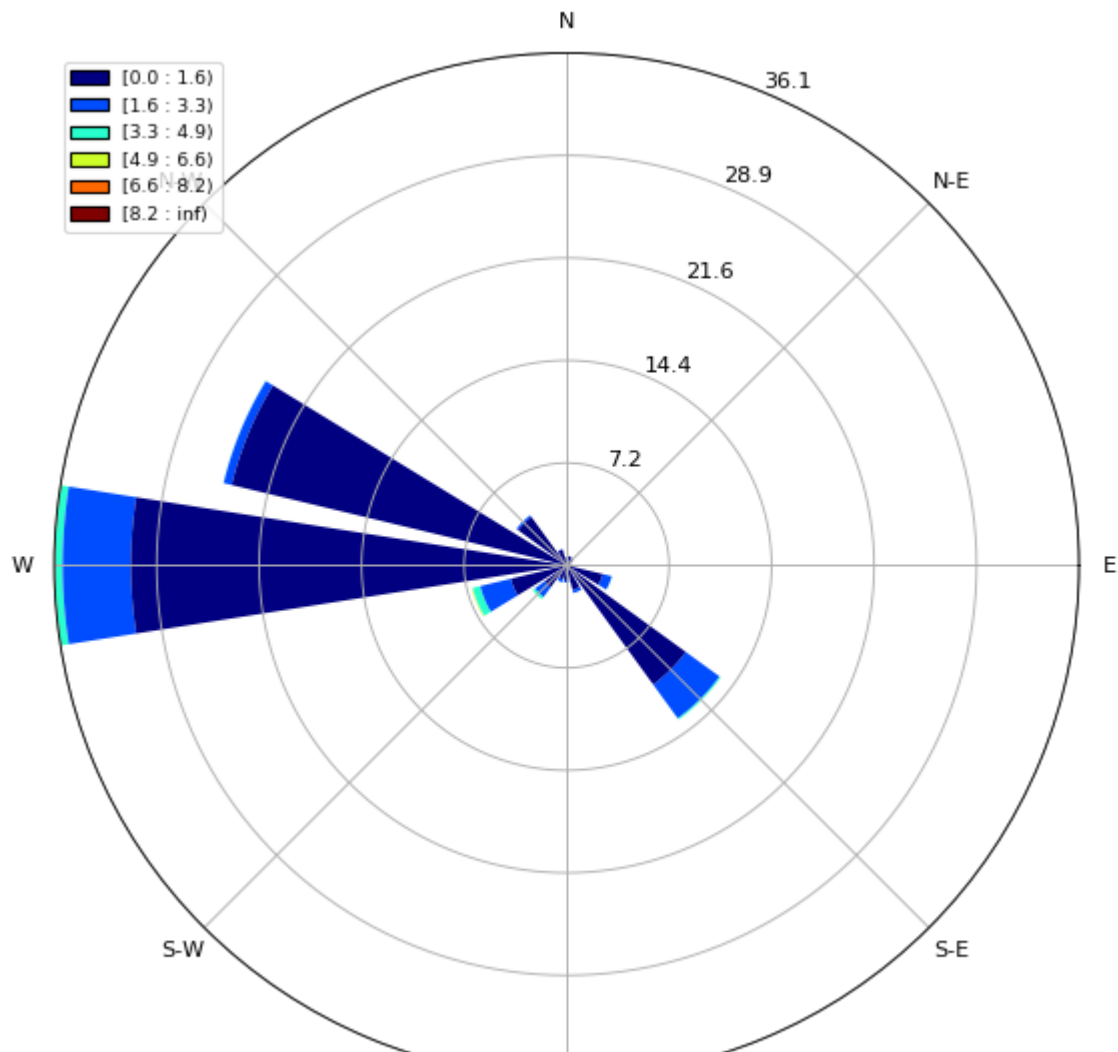
## Activity III.6

Wave rose

```
[86] from windrose import plot_windrose

      ax = plot_windrose(simar, kind='bar', var_name='Hm0', direction_name='DirM')
      ax.legend(loc='upper left', fontsize='small')
```

<matplotlib.legend.Legend at 0x2664fbe0>



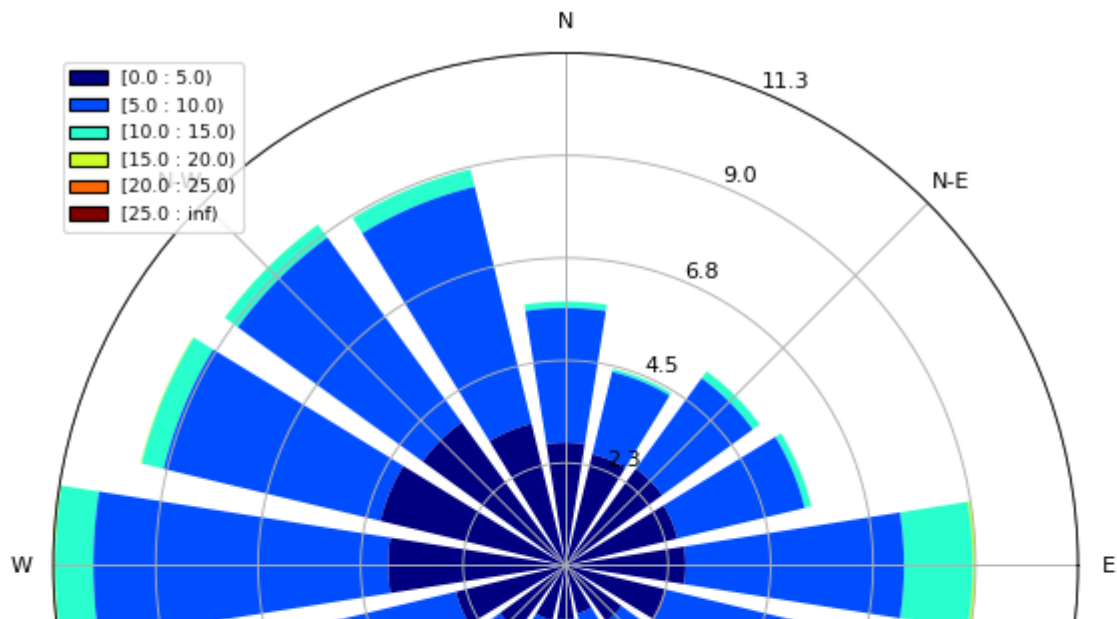
## Exercise III.6

Wind rose

```
[87] from windrose import plot_windrose

ax = plot_windrose(simar, kind='bar', var_name='VelV', direction_name='Di
ax.legend(loc='upper left', fontsize='small')
```

<matplotlib.legend.Legend at 0x26de0ef0>



## Block IV - Find and fill data gaps

Read the file `SIMAR_gaps.txt` using `read_csv`

```
[88] #Read simar gaps
simar_gaps = pd.read_csv('SIMAR_gaps.txt',
                        skiprows=81,
                        delim_whitespace=True,
                        na_values=-99.9,
                        parse_dates=[[0, 1, 2, 3]],
                        index_col=0)

print(simar_gaps)
```

\	Hm0	Tm02	Tp	DirM	Hm0_V	DirM_V	Hm0_F1	Tm02_F1
AA_MM_DD_HH								...
2016-01-01 00:00:00	2.0	8.2	14.1	275.0	NaN	NaN	2.0	8...
2016-01-01 01:00:00	2.0	8.3	13.9	275.0	NaN	NaN	2.0	8...
2016-01-01 02:00:00	2.0	8.1	13.8	275.0	NaN	NaN	2.0	8...
2016-01-01 03:00:00	2.0	7.7	13.7	274.0	NaN	NaN	2.0	8...
2016-01-01 04:00:00	2.0	7.4	13.7	274.0	0.4	224.0	1.9	9...
2016-01-01 05:00:00	1.9	7.2	13.6	273.0	NaN	NaN	1.9	8...
2016-01-01 06:00:00	1.9	7.1	13.6	273.0	NaN	NaN	1.9	8...
2016-01-01 13:00:00	1.9	6.0	13.3	270.0	0.9	242.0	1.6	10...
2016-01-01 14:00:00	1.9	5.7	13.3	268.0	1.0	240.0	1.6	10...
2016-01-01 15:00:00	2.0	5.5	13.3	267.0	1.1	244.0	1.6	10...
2016-01-01 16:00:00	2.0	5.4	13.3	267.0	1.3	247.0	1.5	10...
2016-01-01 17:00:00	2.0	5.4	13.3	267.0	1.4	251.0	1.5	10...
2016-01-01 18:00:00	2.1	5.5	13.3	267.0	1.5	255.0	1.4	11...
2016-01-01 19:00:00	2.1	5.6	13.3	268.0	1.7	260.0	1.2	12...
2016-01-01 20:00:00	2.2	5.6	13.2	267.0	1.8	260.0	1.2	12...
2016-01-01 21:00:00	2.2	5.6	13.2	266.0	1.9	261.0	1.1	12...
2016-01-01 22:00:00	2.3	5.6	13.2	266.0	2.0	261.0	1.1	12...



2016-01-01 23:00:00	2.4	5.6	8.9	266.0	2.1	261.0	1.1	12...
2016-01-03 00:00:00	1.8	9.0	14.1	276.0	NaN	NaN	1.8	9...
2016-01-03 01:00:00	1.8	8.8	13.9	276.0	NaN	NaN	1.8	9...
2016-01-03 02:00:00	1.8	8.6	13.8	275.0	NaN	NaN	1.8	8...
2016-01-03 03:00:00	1.8	8.2	13.7	274.0	NaN	NaN	1.8	8...
2016-01-03 04:00:00	1.7	7.8	13.7	273.0	NaN	NaN	1.7	8...
2016-01-03 05:00:00	1.7	7.5	13.6	273.0	NaN	NaN	1.7	8...
2016-01-03 06:00:00	1.7	7.3	13.5	272.0	NaN	NaN	1.6	8...
2016-01-03 07:00:00	1.7	7.2	13.5	271.0	NaN	NaN	1.6	8...
2016-01-03 08:00:00	1.6	6.9	13.4	271.0	0.4	180.0	1.6	8...
2016-01-03 09:00:00	1.6	6.5	13.4	270.0	NaN	NaN	1.6	8...
2016-01-03 10:00:00	1.7	5.8	13.4	267.0	0.6	202.0	1.5	8...

```
[89] # Find time step
time = simar_gaps.index.to_series().diff()
time_step = stats.mode(time.values)
time_step = pd.to_timedelta(time_step[0])
print(time_step)
```

```
TimedeltaIndex(['01:00:00'], dtype='timedelta64[ns]', freq=None)
```

```
[90] # Reindex to the actual timestep to change gaps for nan
new_index = pd.date_range(simar_gaps.index[0], simar.index[-1], freq=time
simar_gaps_reindex = simar_gaps.reindex(new_index)
```

```
[91] # Interpolate gaps
simar_fill = simar_gaps_reindex.interpolate(method='linear', axis=0, limit=
print(simar_fill)
```

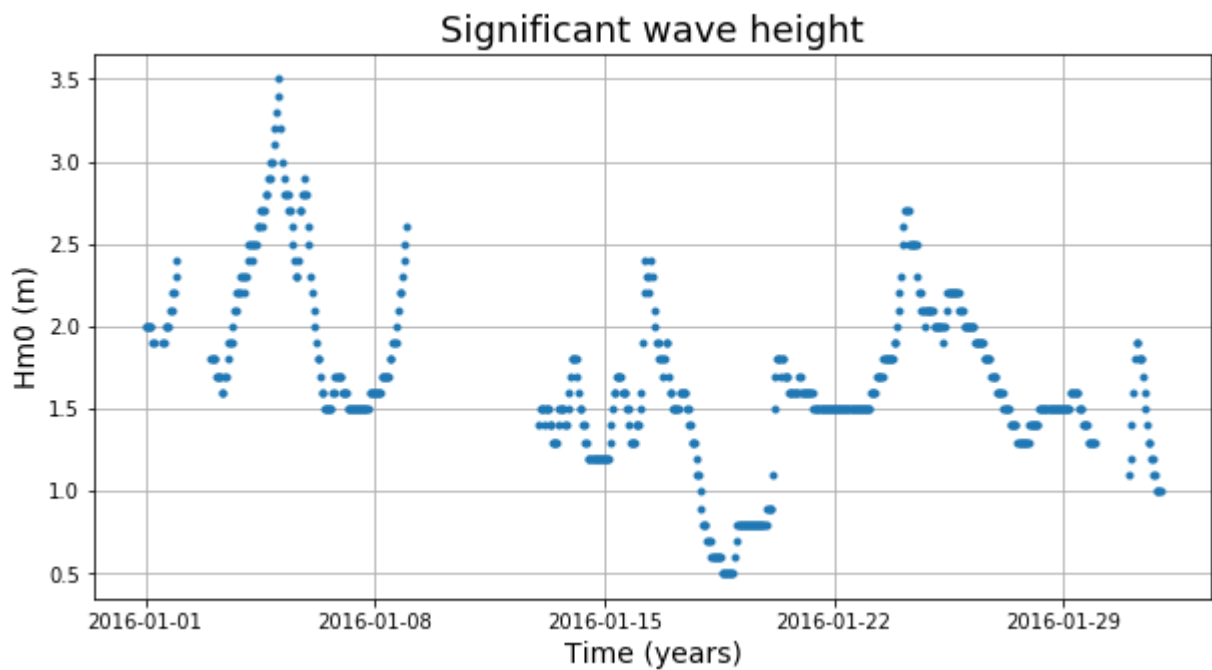
	Hm0	Tm02	Tp	DirM	Hm0_V	\
2016-01-01 00:00:00	2.000	8.200000	14.100000	275.000000	0.400000	
2016-01-01 01:00:00	2.000	8.300000	13.900000	275.000000	0.400000	
2016-01-01 02:00:00	2.000	8.100000	13.800000	275.000000	0.400000	
2016-01-01 03:00:00	2.000	7.700000	13.700000	274.000000	0.400000	
2016-01-01 04:00:00	2.000	7.400000	13.700000	274.000000	0.400000	
2016-01-01 05:00:00	1.900	7.200000	13.600000	273.000000	0.455556	
2016-01-01 06:00:00	1.900	7.100000	13.600000	273.000000	0.511111	
2016-01-01 07:00:00	1.900	6.942857	13.557143	272.571429	0.566667	
2016-01-01 08:00:00	1.900	6.785714	13.514286	272.142857	0.622222	
2016-01-01 09:00:00	1.900	6.628571	13.471429	271.714286	0.677778	
2016-01-01 10:00:00	1.900	6.471429	13.428571	271.285714	0.733333	
2016-01-01 11:00:00	1.900	6.314286	13.385714	270.857143	0.788889	
2016-01-01 12:00:00	1.900	6.157143	13.342857	270.428571	0.844444	
2016-01-01 13:00:00	1.900	6.000000	13.300000	270.000000	0.900000	
2016-01-01 14:00:00	1.900	5.700000	13.300000	268.000000	1.000000	
2016-01-01 15:00:00	2.000	5.500000	13.300000	267.000000	1.100000	
2016-01-01 16:00:00	2.000	5.400000	13.300000	267.000000	1.300000	
2016-01-01 17:00:00	2.000	5.400000	13.300000	267.000000	1.400000	
2016-01-01 18:00:00	2.100	5.500000	13.300000	267.000000	1.500000	
2016-01-01 19:00:00	2.100	5.600000	13.300000	268.000000	1.700000	
2016-01-01 20:00:00	2.200	5.600000	13.200000	267.000000	1.800000	
2016-01-01 21:00:00	2.200	5.600000	13.200000	266.000000	1.900000	
2016-01-01 22:00:00	2.300	5.600000	13.200000	266.000000	2.000000	
2016-01-01 23:00:00	2.400	5.600000	8.900000	266.000000	2.100000	
2016-01-02 00:00:00	2.376	5.736000	9.108000	266.400000	2.048485	

2016-01-02 01:00:00	2.352	5.872000	9.316000	266.800000	1.996970
2016-01-02 02:00:00	2.328	6.008000	9.524000	267.200000	1.945455
2016-01-02 03:00:00	2.304	6.144000	9.732000	267.600000	1.893939
2016-01-02 04:00:00	2.280	6.280000	9.940000	268.000000	1.842424
2016-01-02 05:00:00	2.256	6.416000	10.148000	268.400000	1.790909
...	...	...	...	...	...

Represent the original and interpolated Hm0 time series

```
[92] fig = plt.figure(figsize=(10, 5))
      ax = plt.axes()

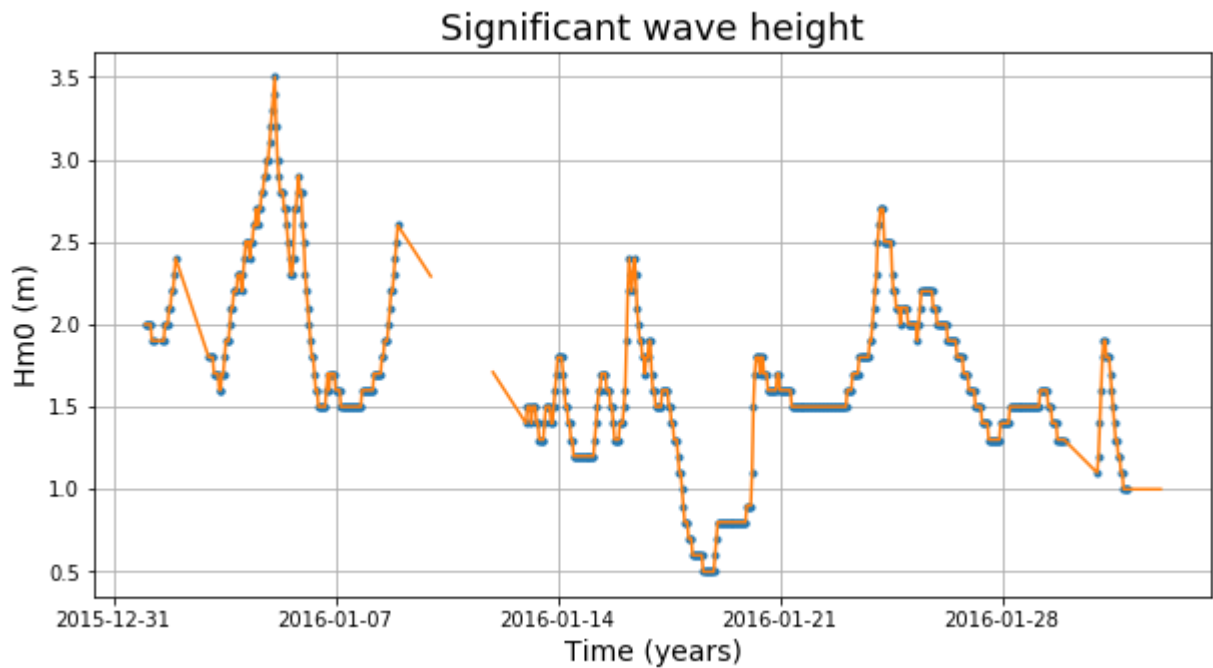
      ax.plot(simar_gaps.loc[:, 'Hm0'], '.')
      ax.set_xlabel('Time (years)', fontsize=14)
      ax.set_ylabel('Hm0 (m)', fontsize=14)
      plt.title('Significant wave height', fontsize=18)
      ax.grid()
```



```
[93] fig = plt.figure(figsize=(10, 5))
      ax = plt.axes()

      ax.plot(simar_gaps.loc[:, 'Hm0'], '.')
      ax.set_xlabel('Time (years)', fontsize=14)
      ax.set_ylabel('Hm0 (m)', fontsize=14)
      plt.title('Significant wave height', fontsize=18)
      ax.grid()
      ax.plot(simar_fill.loc[:, 'Hm0'])
```

[<matplotlib.lines.Line2D at 0x1d75bcc0>]



```
[94] simar_fill = simar_gaps_reindex.interpolate(method='linear', axis=0, limit=10)
```

Represent the original and interpolated Hm0 time series

```
[95] fig = plt.figure(figsize=(10, 5))
      ax = plt.axes()

      ax.plot(simar_gaps.loc[:, 'Hm0'], '.')
      ax.set_xlabel('Time (years)', fontsize=14)
      ax.set_ylabel('Hm0 (m)', fontsize=14)
      plt.title('Significant wave height', fontsize=18)
      ax.grid()
      ax.plot(simar_fill.loc[:, 'Hm0'])
```

```
[<matplotlib.lines.Line2D at 0x220b6160>]
```

Significant wave height

