

```
[1] %matplotlib inline
```

# Análisis de oleaje mediante método de pasos ascendentes por cero

Dept. Mecánica de Estructuras e Ingeniería Hidráulica

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

Universidad de Granada

```
[2] import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

## Parte I

Leer los datos de superficie libre registrados en el laboratorio. Fichero 0093log2.dat

```
[3] datos = pd.read_csv('0093log2.dat',
                      delimiter=',',
                      names=['t', 's1', 's2', 's3', 's4', 's5',
                             's6', 's7', 's8', 's9', 's10'])

datos
```

	t	s1	s2	s3	s4	s5	
0	0.00	0.032479	0.060928	0.054335	0.073871	-0.026129	
1	0.05	-0.040781	0.060928	0.054335	0.049451	-0.001709	
2	0.10	0.056899	0.060928	0.078755	0.049451	-0.001709	
3	0.15	-0.016361	0.060928	0.078755	0.073871	-0.026129	
4	0.20	0.056899	0.012088	0.054335	0.073871	-0.050549	
5	0.25	0.056899	0.060928	0.054335	0.073871	-0.001709	
6	0.30	0.008059	0.085348	0.054335	0.073871	0.022711	
7	0.35	0.130159	0.060928	0.054335	0.073871	-0.026129	

	t	s1	s2	s3	s4	s5
8	0.40	0.056899	0.109768	0.054335	0.073871	-0.050549
9	0.45	0.032479	0.134188	0.054335	0.073871	0.022711
10	0.50	0.056899	0.060928	0.029915	0.073871	-0.026129
11	0.55	0.056899	0.060928	0.054335	0.073871	0.071551
12	0.60	0.301099	0.060928	0.078755	0.073871	-0.026129
13	0.65	0.105739	0.060928	0.054335	0.073871	-0.026129

## Parte 2

En este archivo, la **primera columna es el tiempo** en el que se toma cada dato y **las siguientes contienen los datos de cada uno de los sensores empleados en el ensayo experimental**.

Almacenar en una variable la primera columna para obtener el vector de tiempos.

```
[4] dt = datos.loc[:, 't']
dt = datos.iloc[:, 0]
dt
```

```
0      0.00
1      0.05
2      0.10
3      0.15
4      0.20
5      0.25
6      0.30
7      0.35
8      0.40
9      0.45
10     0.50
11     0.55
12     0.60
13     0.65
14     0.70
15     0.75
16     0.80
17     0.85
18     0.90
19     0.95
20     1.00
21     1.05
22     1.10
23     1.15
24     1.20
25     1.25
26     1.30
27     1.35
28     1.40
```

```
29      1.45
...
1470    73.50
1471    73.55
1472    73.60
```

## Parte 3

Comenzaremos con el **análisis de un único sensor**. Para ello, se debe **seleccionar la columna correspondiente al sensor deseado** y almacenarla en una nueva variable.

```
[5] elev = datos.loc[:, 's1']
elev
```

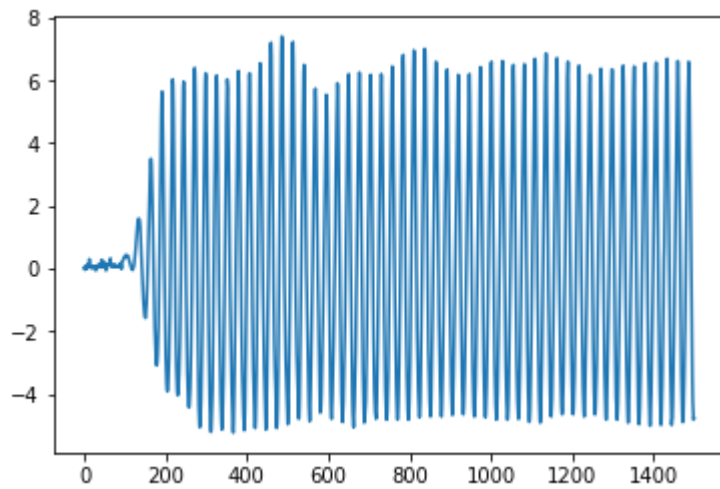
```
0      0.032479
1     -0.040781
2      0.056899
3     -0.016361
4      0.056899
5      0.056899
6      0.008059
7      0.130159
8      0.056899
9      0.032479
10     0.056899
11     0.056899
12     0.301099
13     0.105739
14     0.081319
15     0.056899
16     0.056899
17     0.008059
18     0.056899
19     0.056899
20     0.081319
21     0.032479
22     0.056899
23     0.008059
24     0.056899
25     0.056899
26     0.081319
27     0.056899
28    -0.065201
29     0.056899
...
1470   -4.558486
1471   -4.900367
1472   -4.900367
```

## Parte 4

Procedemos a representar los datos de superficie libre del sensor seleccionado.

```
[6] # Usando pandas
elev.plot()
```

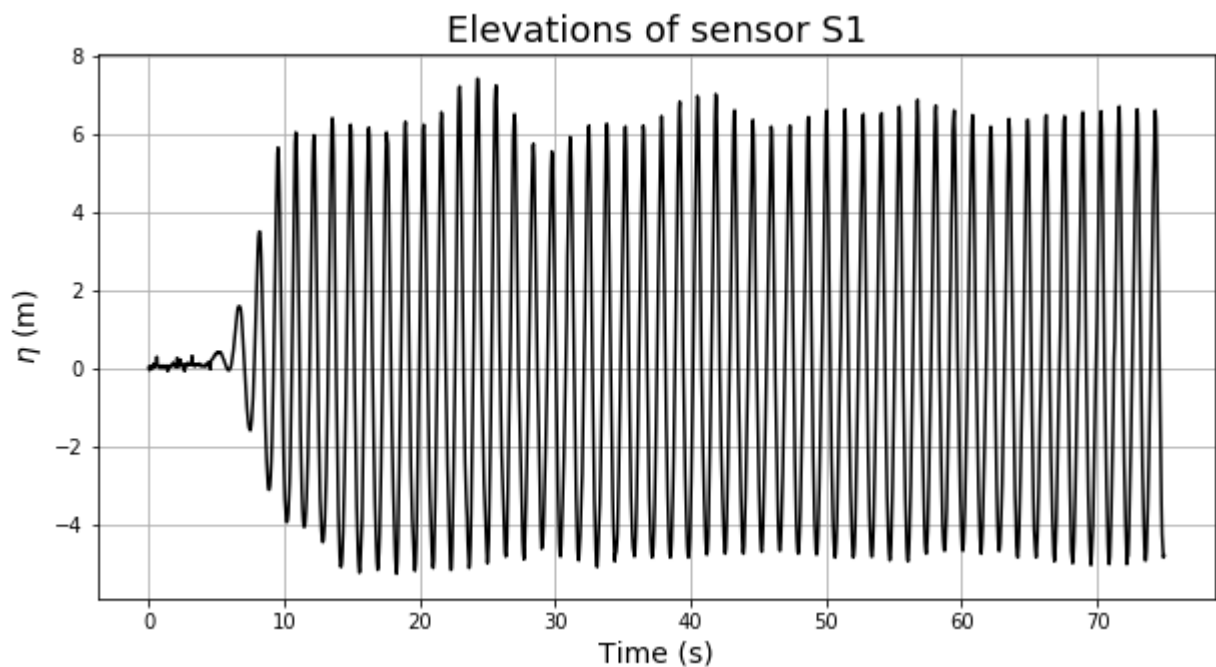
<matplotlib.axes.\_subplots.AxesSubplot at 0xc4b8da0>



```
[7] # Usando matplotlib
fig = plt.figure(figsize=(10, 5))
ax = plt.axes()

ax.plot(dt, elev, 'k')
ax.set_xlabel('Time (s)', fontsize=14)
ax.set_ylabel('$\eta$ (m)', fontsize=14)
plt.title('Elevations of sensor S1', fontsize=18)
ax.grid()

plt.show()
```



## Parte 5

Se deben eliminar los primeros datos correspondientes al inicio del ensayo. Para ello, seleccionaremos el tramo de datos a partir de los 20 segundos:

```
[8] pos = np.where(dt >= 20)
    pos
```

```
(array([ 400,  401,  402, ..., 1497, 1498, 1499], dtype=int64),)
```

```
[9] dt_crop = dt.iloc[pos[0]]
    print(dt_crop)
```

```
400      20.00
401      20.05
402      20.10
403      20.15
404      20.20
405      20.25
406      20.30
407      20.35
408      20.40
409      20.45
410      20.50
411      20.55
412      20.60
413      20.65
414      20.70
415      20.75
416      20.80
417      20.85
418      20.90
419      20.95
420      21.00
421      21.05
422      21.10
423      21.15
424      21.20
425      21.25
426      21.30
427      21.35
428      21.40
429      21.45
...
1470     73.50
1471     73.55
```

```
[10] elev_crop = elev.iloc[pos[0]]
    print(elev_crop)
```

```
400      0.911600
401      1.961661
402      3.060562
403      4.330403
404      5.331624
405      6.113065
```

```

406      6.235165
407      5.868865
408      4.892064
409      3.597802
410      2.181441
411      0.862759
412     -0.187302
413     -1.335043
414     -2.385104
415     -3.190965
416     -3.703785
417     -4.192186
418     -4.680586
419     -5.046887
420     -5.095727
421     -4.900367
422     -4.192186
423     -3.288645
424     -2.238584
425     -1.090843
426      0.056899
427      1.302320
428      2.718682
429      4.037363
      ...
1470    -4.558486
1471      4.000367

```

## Parte 6

Representamos la serie antigua junto con la nueva recortada para verificar que lo hemos hecho bien

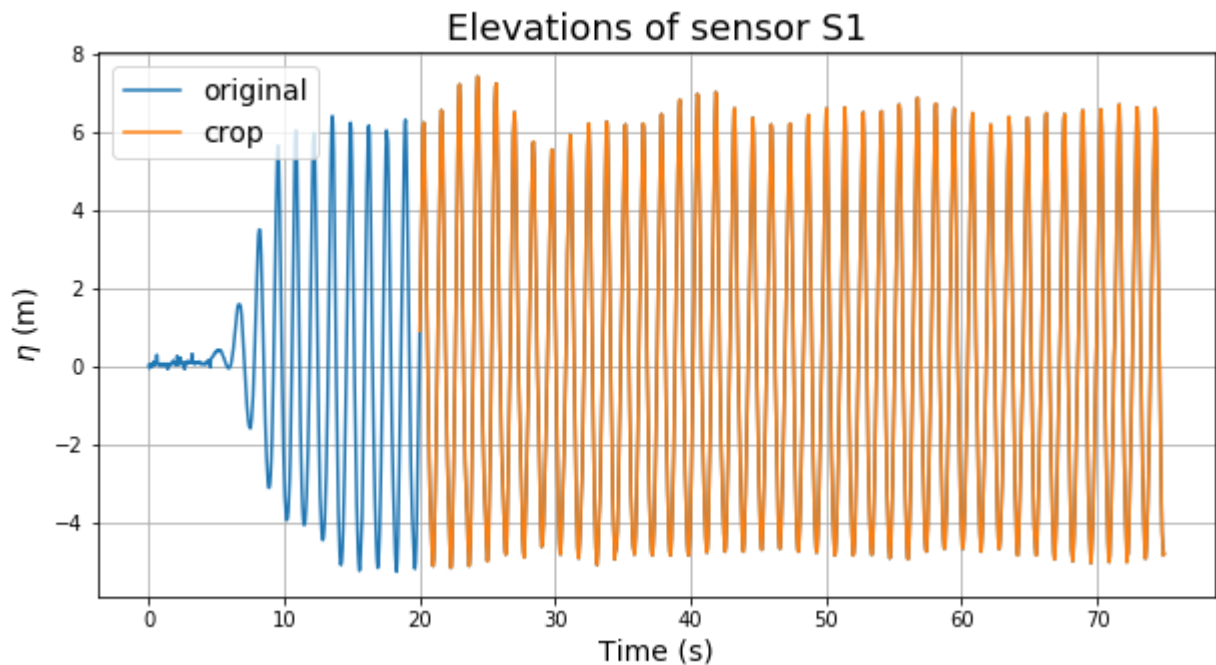
```

[11]  # Usando matplotlib
      fig = plt.figure(figsize=(10, 5))
      ax = plt.axes()

      ax.plot(dt, elev)
      ax.plot(dt_crop, elev_crop)
      ax.set_xlabel('Time (s)', fontsize=14)
      ax.set_ylabel('$\eta$ (m)', fontsize=14)
      plt.title('Elevations of sensor S1', fontsize=18)
      ax.grid()
      ax.legend(['original', 'crop'], fontsize=14)

      plt.show()

```



## Parte 7

El método de pasos ascendentes por cero consiste en buscar las líneas en las que el valor de la superficie libre del agua pasa de negativo a positivo. Almacene en una lista las posiciones de los pasos ascendentes.

```
[12] num_datos = np.size(elev_crop)
pos_p_asc = []
for i in range(0, num_datos-1):

    if elev_crop.iloc[i] <= 0.0 and elev_crop.iloc[i+1] > 0.0:
        pos_p_asc.append(i)
print(pos_p_asc)
```

```
[25, 52, 79, 105, 133, 161, 188, 215, 242, 269, 296, 323, 350, 376, 403,
430, 457, 484, 511, 539, 566, 593, 620, 647, 674, 700, 727, 754, 781, 808,
836, 863, 890, 917, 944, 971, 998, 1024, 1051, 1078]
```

## Parte 8

Representamos las posiciones

```
[13] # Usando matplotlib
fig = plt.figure(figsize=(10, 5))
ax = plt.axes()

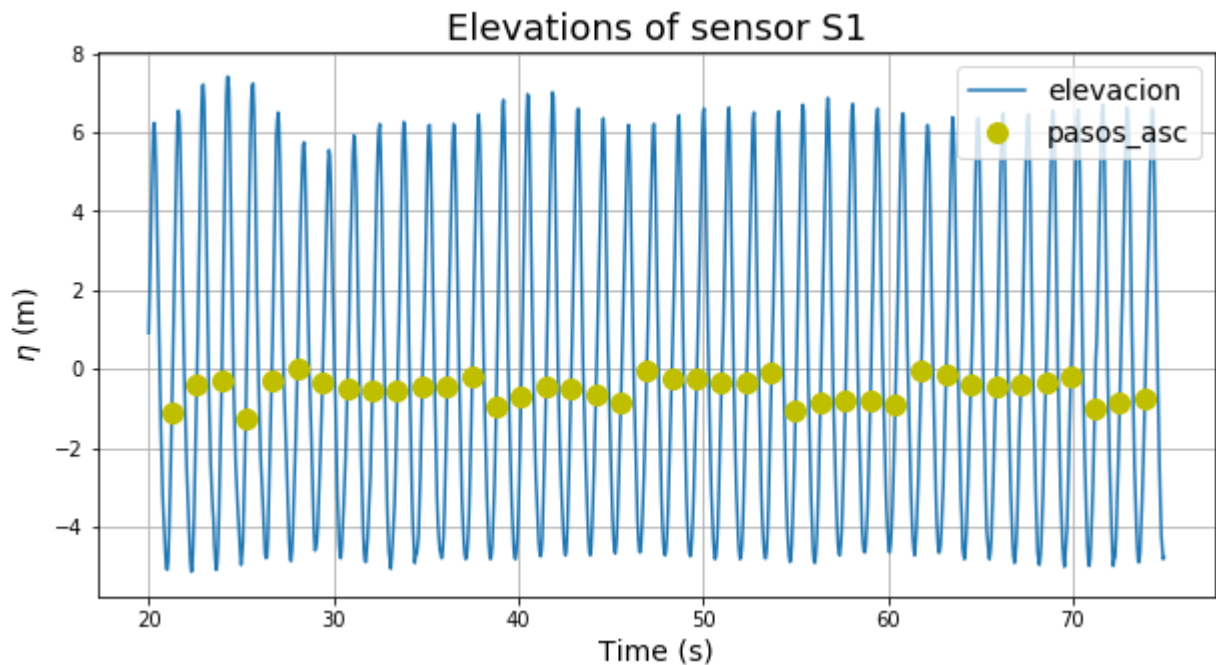
ax.plot(dt_crop, elev_crop)
ax.plot(dt_crop.iloc[pos_p_asc], elev_crop.iloc[pos_p_asc], '.y', markers
ax.set_xlabel('Time (s)', fontsize=14)
```

```

ax.set_ylabel('$\eta$ (m)', fontsize=14)
plt.title('Elevations of sensor S1', fontsize=18)
ax.grid()
ax.legend(['elevacion', 'pasos_asc'], fontsize=14)

plt.show()

```



## Parte 9

**Entre cada dos pasos ascendentes por cero tenemos una ola.** A continuación calcular **para cada ola su altura y su periodo:**

```

[14] num_pasos_asc=np.size(pos_p_asc)

# Create an empty DataFrame
data_ht = pd.DataFrame(np.nan, index=range(num_pasos_asc - 1), columns=range(2))

for j in range(0, num_pasos_asc - 1):
    #Elevacion entre pasos ascendentes
    elev_crop_p_asc = elev_crop.iloc[pos_p_asc[j]: pos_p_asc[j+1]]

    # Altura de ola
    alt_ola = np.max(elev_crop_p_asc)-np.min(elev_crop_p_asc)

    # Periodo
    periodo = dt_crop.iloc[pos_p_asc[j+1]] - dt_crop.iloc[pos_p_asc[j]]

    data_ht.iloc[j, 0] = alt_ola
    data_ht.iloc[j, 1] = periodo

print(data_ht)

```



	0	1
0	11.697192	1.35
1	12.307693	1.35
2	12.380953	1.30
3	12.039073	1.40
4	11.379732	1.40
5	10.354092	1.35
6	10.354091	1.35
7	10.818072	1.35
8	11.282052	1.35
9	11.184372	1.35
10	10.989012	1.35
11	11.037851	1.35
12	11.282051	1.30
13	11.648352	1.35
14	11.721613	1.35
15	11.746032	1.35
16	11.330891	1.35
17	11.037851	1.35
18	10.842492	1.40
19	10.940171	1.35
20	11.184372	1.35
21	11.428571	1.35
22	11.452991	1.35
23	11.306472	1.35
24	11.428572	1.30
25	11.623932	1.35
26	11.599512	1.35
27	11.379733	1.35
28	11.257632	1.35
29	11.208791	1.40
30	10.842492	1.35
31	11.208791	1.35

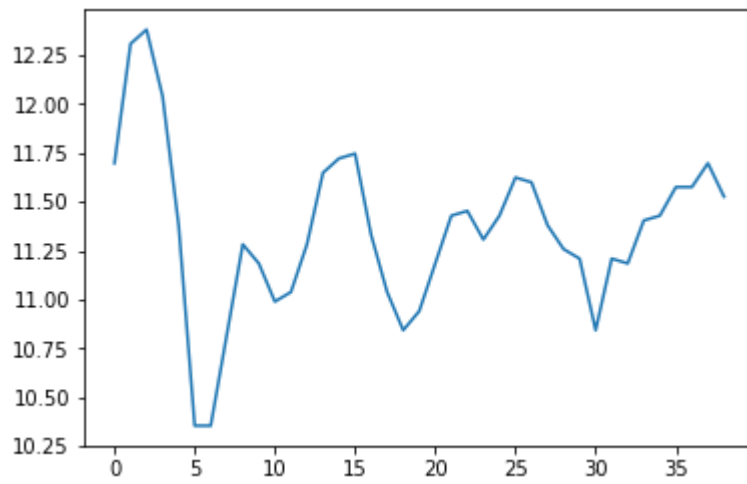
## Parte 10

Por último, guardar los datos de H-T a un archivo para su posterior análisis.

```
[18] data_ht.to_csv('HT.dat', float_format='%g', index=False, header=['H', 'T'])
```

```
[16] data_ht.iloc[:, 0].plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0xcc4a9e8>
```



```
[17] data_ht.iloc[:, 1].plot()
```

<matplotlib.axes.\_subplots.AxesSubplot at 0xd90f860>

