"El saber de mis hijos hará mi grandeza"

Universidad de Sonora
Departamento de Física
Licenciatura en Física
Física Computacional-1
2016-2

Actividad 6: Buscando regularidades en las Mareas

Danira Rios Quijada
Profesor: Carlos Lizárraga Celaya

29 de Octubre de 2016

Resumen

En el siguiente reporte de trabajo, describo los pasos que lleve a cabo para lograr graficar datos de nivel de mar de la Isla Tiburón y del Manglar Sargento, para dar inicio a la actividad 6 de Física Computacional; se muestran gráficas por día, semana y mes de cada conjunto de datos. Existen algunas dificultades, dado que los datos de la Isla Tiburón son pronósticos del 2016 y los datos de Sargento son registros de 2013-2014, además el intervalo entre mediciones no coincide, esto será solucionado en contribuciones posteriores a ésta actividad.

1. Introducción [3]

El objetivo de esta actividad, y tal vez de la siguiente es poder hacer un análisis armónico de las mareas; La teoría de las mareas se ha estudiado matemáticamente desde los tiempos de Isaac Newton quien desarrolló la teoría del equilibrio de mareas,como complemento a esta teoría, Laplace desarrolló la teoría dinámica de mareas, la cual conciste en analizar y poder hacer predicciones acerca de las mareas a partir de series de Fourier, lo que es prácticamente un aálisis espectral o descomposición en frecuencias de un fenómeno físico.

2. Datos Isla Tiburón

2.1. Descargando los datos del año 2016

El archivo de datos que descargamos, lo descargamos directamente de la página web de pronósticos del CICESE[2], decidí descargar los datos de la isla tiburón, el archivo es csv, y lo nombre "tiburon2016.csv", consistió en un archivo con 5 columnas, 4 de ellas dedicadas a la fecha y hora de la medición y una para la altura (mm), pronosticada.

2.2. Dando formato al archivo csv

El siguiente paso fue darle formato al archivo original, lo primero fue juntar los datos de la fecha y hora en una sola columna y la vez darles el formato adecuado, (de fecha y hora), para lo cual utilizamos el siguiente codigo:

```
from datetime import datetime
df['date']= df.apply(lambda x:datetime.strptime("{0} {1} {2} {3}"
.format(x[u'anio'],x[u'mes'], x[u'dia'], x[u'hora(utc)']),
"%Y %m %d %H"),axis=1)
```

Donde se crean una nueva columna con las columnas de anio, mes día y hora, y a su vez se le da el formato año, mes, día y hora, obteniendo algo como esto:

	anio	mes	dia	hora(utc)	altura(mm)	date
0	2016	1	1	0	979	2016-01-01 00:00:00
1	2016	1	1	1	982	2016-01-01 01:00:00
2	2016	1	1	2	948	2016-01-01 02:00:00
3	2016	1	1	3	889	2016-01-01 03:00:00
4	2016	1	1	4	820	2016-01-01 04:00:00

Ahora para suprimir las columnas que no necesitamos y acomadar la nueva columna, utilizamos el siguiente codigo:

```
df_short=df.drop(df.columns[[0,1,2,3]],axis=1)
```

Suprimiendo las columnas 0,1,2 y 3. Después para acomodar la columna "date", utilizamos:

```
cols=df_short.columns.tolist()
cols=cols[-1:] + cols [:-1]
df_short=pd.DataFrame(df_short[cols])
```

Obteniendo algo como lo siguiente:

	date	altura(mm)
0	2016-01-01 00:00:00	979
1	2016-01-01 01:00:00	982
2	2016-01-01 02:00:00	948
3	2016-01-01 03:00:00	889
4	2016-01-01 04:00:00	820
5	2016-01-01 05:00:00	761
6	2016-01-01 06:00:00	736
7	2016-01-01 07:00:00	757

Aún en este nuevo formato de los datos, había algunas casillas NaN, lo cual no nos permitía graficar, dado que no podiamos eliminar las filas con datos faltantes porque tenía que haber una secuencia de tiempo, por lo que obtamos por el método de interpolación de segundo grado, lo que significa que usando los datos existentes creamos datos para las casillas vacías, de tal manera que estos datos no alteraran la estadística de nuestros datos, utilizamos el siguiente código:

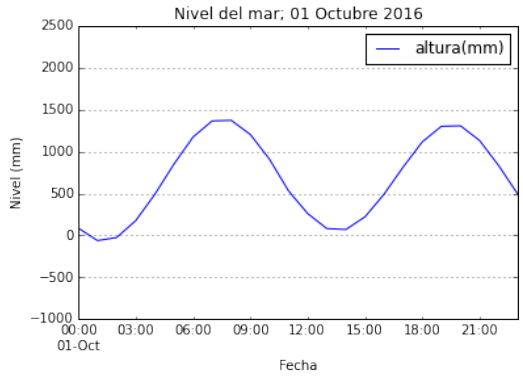
```
import scipy
df_short['altura(mm)']=
df_short['altura(mm)'].astype(float).interpolate
(method='spline', order=2)
```

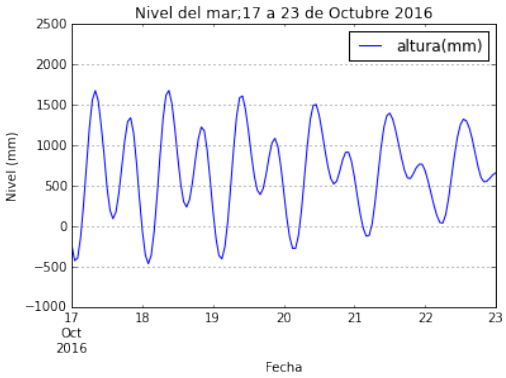
Una vez que hicimos la interpolación, proseguimos a graficar los datos, sin embargo antes de graficar pasamos los datos de la columna "date.ªl index, para facilitar la graficación, una vez hecho esto, pasamos a graficar los datos por mes, semana y día, con el siguiente código:

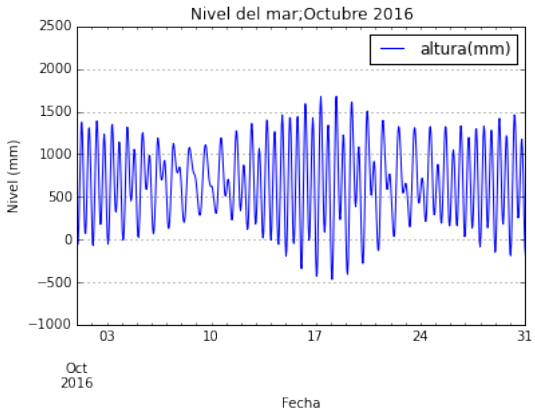
```
import pandas as pd
import numpy as np
import matplotlib as plt
import matplotlib.pyplot as mplt

ax=df_short.plot()
ax.set_xlim(pd.Timestamp('2016-10-01 00:00:00'),
pd.Timestamp('2016-10-01 23:00:00'))
mplt.ylabel('Nivel (mm)')
mplt.xlabel('Fecha')
mplt.title('Nivel del mar; 01 Octubre 2016')
```

Donde los límites de graficación se definen en pd.Timestamp(); estos fueron los resultados obtenidos:







3. Datos Sargento

3.1. Dando formato al archivo

El archivo original consistía en un archivo excel, con varias columnas adicionales, y datos desde Octubre de 2013 a Marzo de 2014; además había varios breaks en el formato de la fecha, pero como nuestro interés era solo graficar un día, un mes y una semana, obté por solo trabajar con el primer conjunto de datos con el mismo formato de fecha; octubre-diciembre 2013.
Primeramente leí el archivo con excel y después lo convertí a csv, para poder trabajar con él en pandas, eliminé las columnas adicionales, obteniendo:

	Date	Time	Water Level
0	10/13/2013	15:30	-0.097
1	10/13/2013	16:00	-0.104
2	10/13/2013	16:30	-0.109
3	10/13/2013	17:00	-0.112
4	10/13/2013	17:30	-0.117

Igual que en el caso anterior, tenemos dos columnas con los datos de fecha y hora, la solución fue crear una nueva columna juntando las columnas de Date y Time, con el siguiente código:

```
df['fecha'] = df[u'Date']+ ' ' + df[u'Time']
```

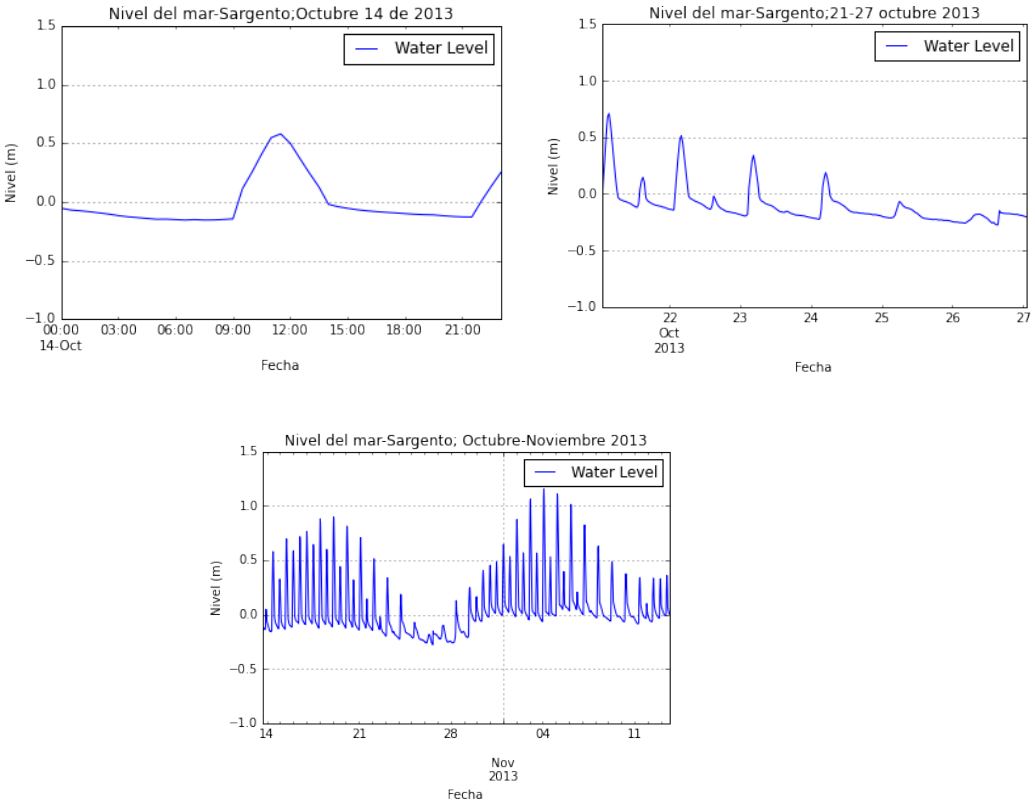
Después le dimos a esta columna formato de fecha y hora, con el siguiente código:

```
df['date']=pd.to_datetime(df['fecha'], format='%m/%d/%Y %H:%M')
```

Suprimimos las columnas extras, y situamos la columna "date.[en] el index obteniendo algo como lo siguiente:

	Water Level
date	
2013-10-13 15:30:00	-0.097
2013-10-13 16:00:00	-0.104
2013-10-13 16:30:00	-0.109
2013-10-13 17:00:00	-0.112
2013-10-13 17:30:00	-0.117
2013-10-13 18:00:00	-0.121
2013-10-13 18:30:00	-0.123
2013-10-13 19:00:00	-0.127

A partir de lo anterior, pudimos graficar; un día, una semana y un mes, con los siguientes resultados:



4. Bibliografía

Referencias

[1] Carlos Lizárraga Celaya, *Actividad seis, curso de Física Computacional 1*, (2016, 12 de Octubre).Recuperado (2016, 29 de octubre), Desde: `http://computacional1.pbworks.com/w/page/111907588/Actividad20620282016-229`

[2] CICESE *Pronostico de mareas*, (2015), recuperado (2016, 17 de Octubre). Desde: `http://predmar.cicese.mx/calendarios/`

[3] Grupo de Din´amica de Flujos Ambientales, Universidad de Granada, *Análisis de Fourier*, (2014-2015). Recuperado (2016,29 de Octubre), Desde: `http://wdb.ugr.es/~mdiezm/wp-content/uploads/IMCtema04fourier-2014-2015.pdf`