Taller 6

Métodos Computacionales para Políticas Públicas - URosario

Entrega: viernes 27-sep-2019 11:59 PM

Juan Sebastián Muñoz

jsebastianmvargas@gmail.com (mailto:jsebastianmvargas@gmail.com)

Instrucciones:

- Guarde una copia de este *Jupyter Notebook* en su computador, idealmente en una carpeta destinada al material del curso.
- Modifique el nombre del archivo del *notebook*, agregando al final un guión inferior y su nombre y apellido, separados estos últimos por otro guión inferior. Por ejemplo, mi *notebook* se llamaría: mcpp_taller6_santiago_matallana
- Marque el *notebook* con su nombre y e-mail en el bloque verde arriba. Reemplace el texto "[Su nombre acá]" con su nombre y apellido. Similar para su e-mail.
- Desarrolle la totalidad del taller sobre este *notebook*, insertando las celdas que sea necesario debajo de cada pregunta. Haga buen uso de las celdas para código y de las celdas tipo *markdown* según el caso.
- Recuerde salvar periódicamente sus avances.
- Cuando termine el taller:
  1. Descárguelo en PDF. Si tiene algún problema con la conversión, descárguelo en HTML.
  2. Suba todos los archivos a su repositorio en GitHub, en una carpeta destinada exclusivamente para este taller, antes de la fecha y hora límites.

(Todos los ejercicios tienen el mismo valor.)

Resuelva la parte 1 de este documento (http://www.math.pitt.edu/~sussmanm/3040Summer14/exercisesII.pdf).

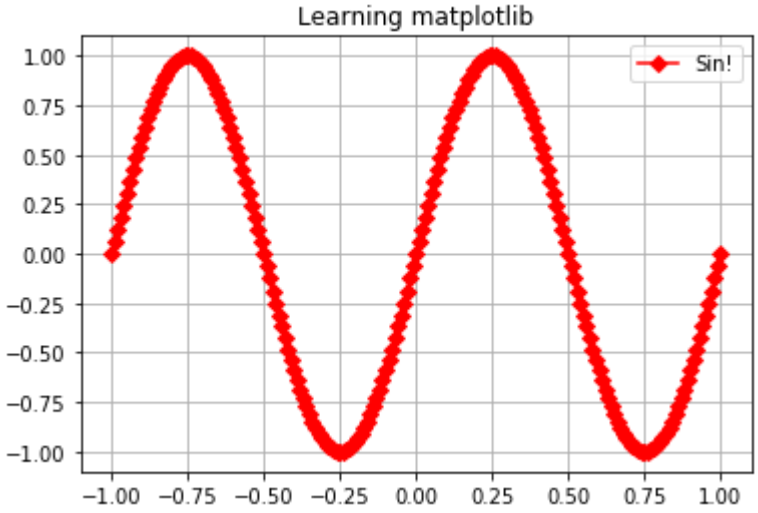
In [2]:
```
import numpy as np
import scipy.linalg as la
import matplotlib.pyplot as plt
import math
#Punto 1
m1 = 40
print ("The value is", m1)
#Punto 2
Square = m1 ** 2
print ("The square of", m1, "is", Square)
Cube = m1 ** 3
print ("The cube of", m1, "is", Cube)
#Punto 3
theta = 99 * (math.pi / 180)
print ("Theta is", theta , "radians")
#Punto 4
Sin = math.sin(theta)
print ("The sin of", theta, "is", Sin)
Cos = math.cos(theta)
print ("The cos of", theta, "is", Cos)
## I'm using theta as radians, due to we multiply by 99 degrees to (pi/180)
## Punto 5
meshPoints = np.linspace(-1, 1, num=500)
## Punto 6
print ("the value of the 53th element is" , meshPoints[52])
## Punto 7
plt.plot(meshPoints,np.sin(2*math.pi*meshPoints));
plt.savefig("sin_plot.png")
```

```
The value is 40
The square of 40 is 1600
The cube of 40 is 64000
Theta is 1.7278759594743862 radians
The sin of 1.7278759594743862 is 0.9876883405951378
The cos of 1.7278759594743862 is -0.1564344650402308
the value of the 53th element is -0.7915831663326653
```

Resuelva los ejercicios de las secciones 4.1, 5.1, 6.1, 7.4 y 8.5 de este documento (http://www.python-academy.com/download/pycon2012/matplotlib_handout.pdf).

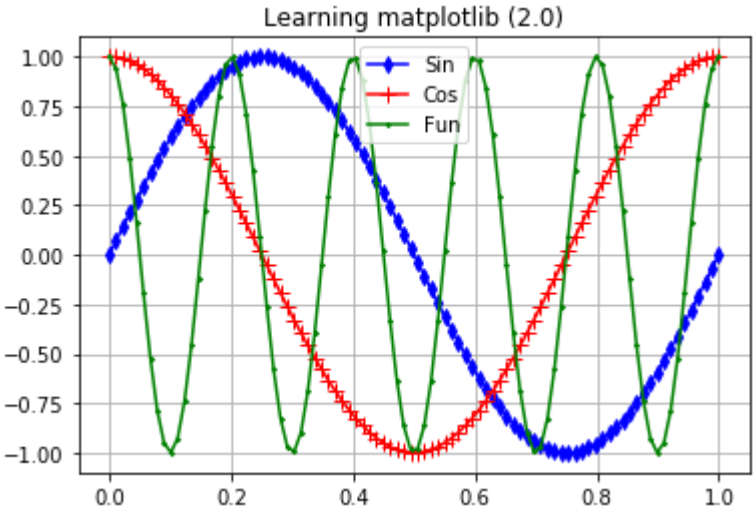
Punto 4.1

In [61]:
```
values = np.linspace(-1, 1, 201)
plt.plot(values,np.sin(2*math.pi*values), "r", marker="D", ms=5, label="Sin!")
plt.legend()
plt.title("Learning matplot")
plt.grid()
plt.show();
```



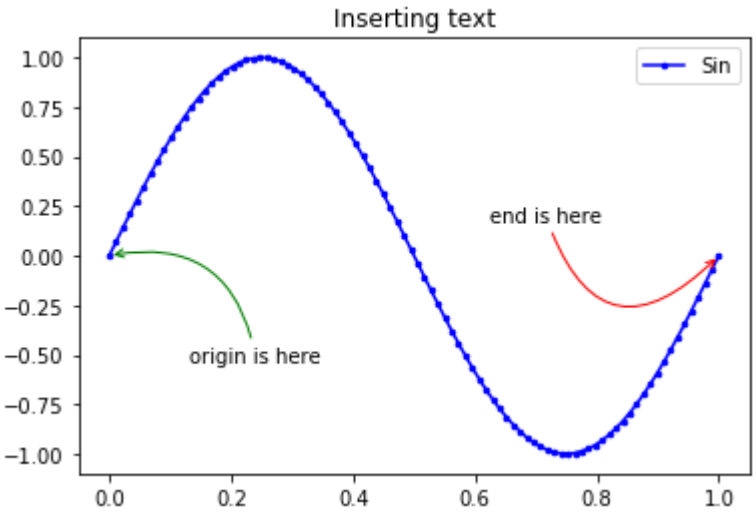
Punto 5.1

In [86]:
```
values = np.linspace(0, 1, 90)
plt.plot(values,np.sin(2*math.pi*values), "b", marker="d", ms=5, label="Sin")
plt.plot(values,np.cos(2*math.pi*values), "r", marker="+", ms=8, label="Cos")
plt.plot(values,np.cos(10*math.pi*values), "g", marker="*", ms=2, label="Fun")
plt.legend()
plt.title("Learning matplotlib (2.0)")
plt.grid()
plt.show();
```



Punto 6.1

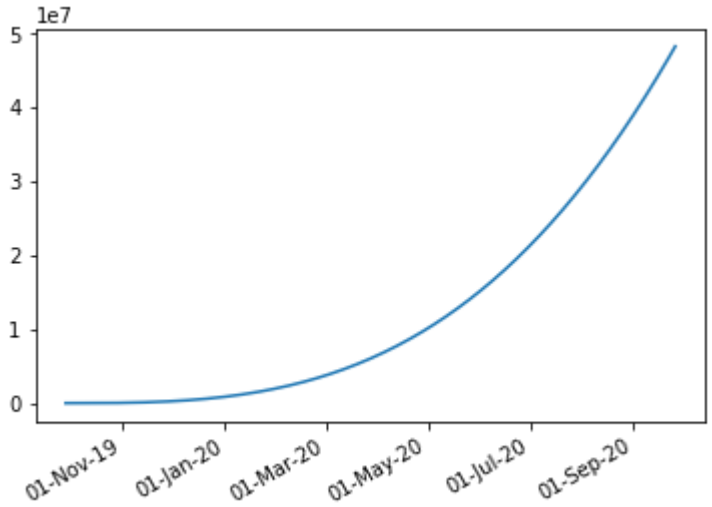
In [163]:
```
values = np.linspace(0, 1, 90)
plt.plot(values,np.sin(2*math.pi*values), "b", marker=".", ms=5, label="Sin")
plt.legend()
plt.title("Annotating text")
#Using Data Coordinates and figure points
plt.annotate('origin is here', (0.0, 0.0), (100, 80) ,
            arrowprops=dict(arrowstyle='->', color='green', connectionstyle='arc
            xycoords='data', textcoords='figure points')

plt.annotate('end is here', (1.0, 0.0), (250, 150) ,
            arrowprops=dict(arrowstyle='->', color='red', connectionstyle='arc3,
            xycoords='data', textcoords='figure points')
plt.show();
```



Punto 7.4

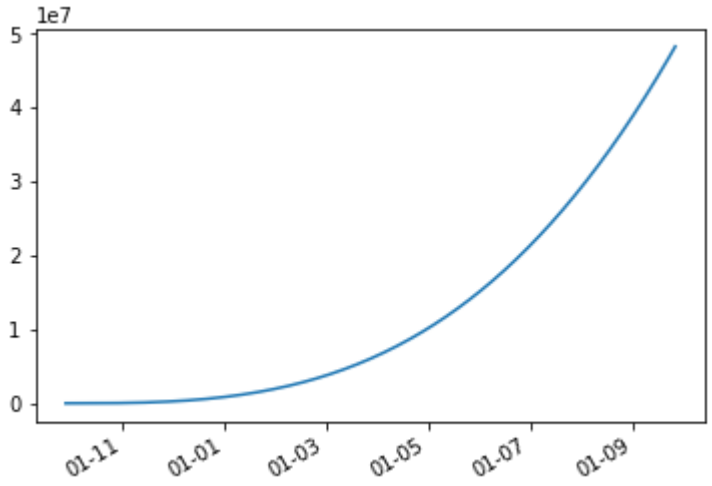
Punto 1 y punto 2

In [199]:
```
import datetime
import random
import matplotlib.dates as mdates
import matplotlib.pyplot as plt
#point 1
x = [datetime.datetime.now() + datetime.timedelta(days=i) for i in range(365)]
y = [i**3+random.gauss(0,5) for i,_ in enumerate(x)]
plt.plot(x,y)
# point 2
#Dates in such a way that only the first day of the month is shown
plt.gcf().autofmt_xdate()
myFmt = mdates.DateFormatter('%d-%b-%y')
plt.gca().xaxis.set_major_formatter(myFmt)
plt.show()
```

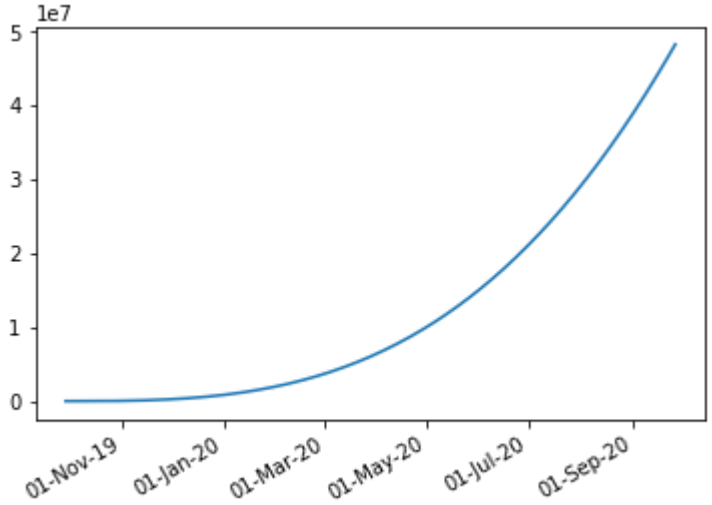


Punto 3

In [228]:
```
#point 3
#Dates without year
import datetime
import random
import matplotlib.dates as mdates
import matplotlib.pyplot as plt
x = [datetime.datetime.now() + datetime.timedelta(days=i) for i in range(365)]
y = [i**3+random.gauss(0,5) for i,_ in enumerate(x)]
plt.plot(x,y)
plt.gcf().autofmt_xdate()
#Month as number
myFmt = mdates.DateFormatter('%d-%m')
plt.gca().xaxis.set_major_formatter(myFmt)
plt.show()
```

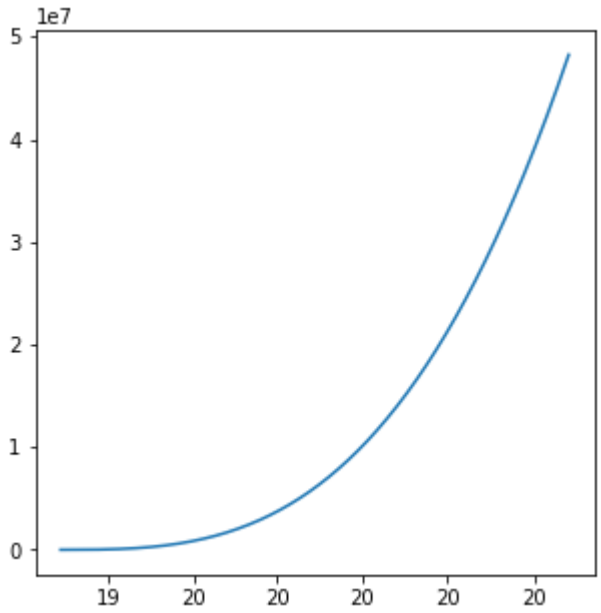


In [3]:
```
#point 3
#Dates with year
import datetime
import random
import matplotlib.dates as mdates
import matplotlib.pyplot as plt
x = [datetime.datetime.now() + datetime.timedelta(days=i) for i in range(365)]
y = [i**3+random.gauss(0,5) for i,_ in enumerate(x)]
plt.plot(x,y)
plt.gcf().autofmt_xdate()
#Month as first three letters of the month name
myFmt = mdates.DateFormatter('%d-%b-%y')
plt.gca().xaxis.set_major_formatter(myFmt)
plt.show()
```

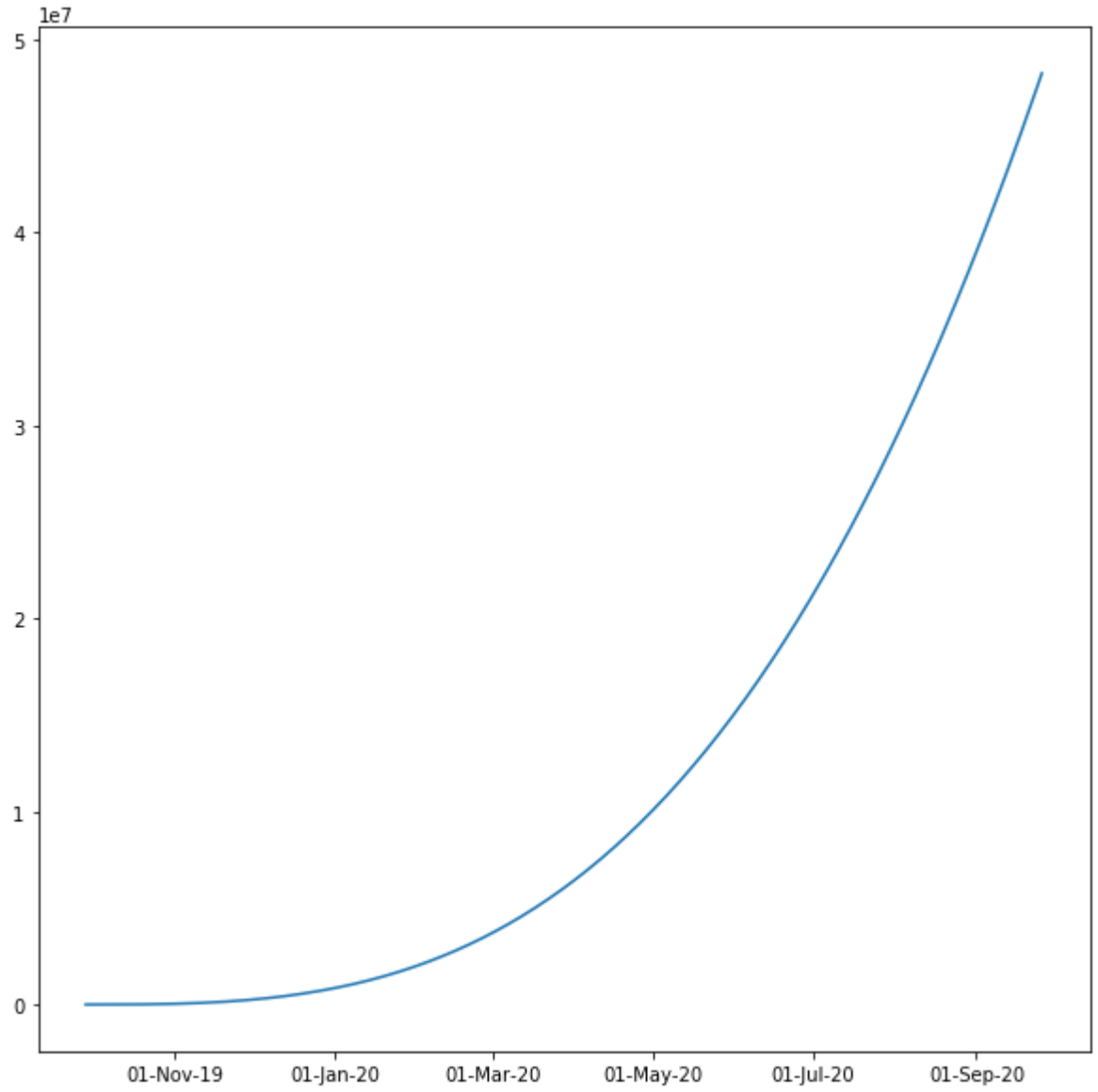


Punto 8.5

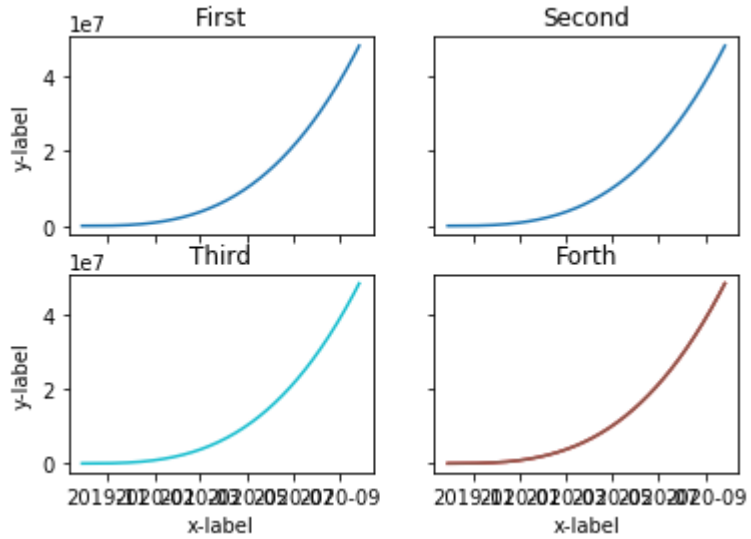
In [46]:
```
#1. Draw two figures, one 5 by 5
plt.figure(figsize=[5,5])
x = [datetime.datetime.now() + datetime.timedelta(days=i) for i in range(365)]
y = [i**3+random.gauss(0,5) for i,_ in enumerate(x)]
myFmt = mdates.DateFormatter('%y')
plt.gca().xaxis.set_major_formatter(myFmt)
plt.plot(x,y);
```



In [48]:
```
#1. Draw two figures, one 10 by 10 inches.
plt.figure(figsize=[10,10])
x = [datetime.datetime.now() + datetime.timedelta(days=i) for i in range(365)]
y = [i**3+random.gauss(0,5) for i,_ in enumerate(x)]
myFmt = mdates.DateFormatter('%d-%b-%y')
plt.gca().xaxis.set_major_formatter(myFmt)
plt.plot(x,y);
```



In [52]:
```
#2. Add four subplots to one figure. Add labels and ticks only to the outermost a

fig, axs = plt.subplots(2, 2)
axs[0, 0].plot(x, y)
axs[0, 0].set_title('First')
axs[0, 1].plot(x, y, 'tab:blue')
axs[0, 1].set_title('Second')
axs[1, 0].plot(x, y, 'tab:cyan')
axs[1, 0].set_title('Third')
axs[1, 1].plot(x, y, 'tab:red')
axs[1, 1].set_title('Forth')
axs[1, 1].plot(x, y, 'tab:brown')
for ax in axs.flat:
    ax.set(xlabel='x-label', ylabel='y-label')

# Adding labels and ticks only to the outermost axes.
for ax in axs.flat:
    ax.label_outer()
```



In [26]:
```
#3. Place a small plot in one bigger plot.
ax1 = plt.axes()  # standard axes
ax2 = plt.axes([0.40, 0.40, 0.2, 0.2])
```