2019_1_mcpp_taller_6_mateo_lancheros

March 29, 2019

1 Taller 6

Métodos Computacionales para Políticas Públicas - URosario

Entrega: viernes 29-mar-2019 11:59 PM

César Mateo Lancheros Cañón cesar.lancheros@urosario.edu.co

1.1 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.)

1.1.1 Resuelva la parte 1 de este documento.

1.2 Exercises on numpy, scipy, and matplotlib

1.2.1 1. Exercise 7: Numpy practice (5 points)

Start up Python and use it to answer the following ques- tions. Use the following imports:

```
In [1]: import numpy as np
        import scipy.linalg as la
        import matplotlib.pyplot as plt
        import math
        math.pi
```

```
Out[1]: 3.141592653589793
```

1. Choose a value and set the variable x to that value.

```
In [2]: x = 20
```

2. What is command to compute the square of x? Its cube?

```
In [3]: x ** 2

Out[3]: 400

In [4]: x ** 3

Out[4]: 8000
```

3. Choose an angle and set the variable theta to its value (a number).

```
In [5]: theta = 0.45
```

4. What is sin ? cos ? Angles can be measured in degrees or radians. Which of these are being used used?

```
In [6]: np.sin(theta)

Out[6]: 0.43496553411123023

In [7]: np.cos(theta)

Out[7]: 0.9004471023526769
```

-Para este caso, estan siendo usados radians

5. Use the np.linspace function to create a row vector called meshPoints containing exactly 500 values with values evenly spaced between -1 and 1.

```
In [8]: meshPoints = np.linspace(-1,1,500)
        print (meshPoints)

[-1.         -0.99599198 -0.99198397 -0.98797595 -0.98396794 -0.97995992
 -0.9759519  -0.97194389 -0.96793587 -0.96392786 -0.95991984 -0.95591182
 -0.95190381 -0.94789579 -0.94388778 -0.93987976 -0.93587174 -0.93186373
 -0.92785571 -0.9238477  -0.91983968 -0.91583166 -0.91182365 -0.90781563
 -0.90380762 -0.8997996  -0.89579158 -0.89178357 -0.88777555 -0.88376754
 -0.87975952 -0.8757515  -0.87174349 -0.86773547 -0.86372745 -0.85971944
 -0.85571142 -0.85170341 -0.84769539 -0.84368737 -0.83967936 -0.83567134
 -0.83166333 -0.82765531 -0.82364729 -0.81963928 -0.81563126 -0.81162325
 -0.80761523 -0.80360721 -0.7995992  -0.79559118 -0.79158317 -0.78757515
 -0.78356713 -0.77955912 -0.7755511  -0.77154309 -0.76753507 -0.76352705
 -0.75951904 -0.75551102 -0.75150301 -0.74749499 -0.74348697 -0.73947896
 -0.73547094 -0.73146293 -0.72745491 -0.72344689 -0.71943888 -0.71543086
```

```
-0.71142285 -0.70741483 -0.70340681 -0.6993988  -0.69539078 -0.69138277
-0.68737475 -0.68336673 -0.67935872 -0.6753507  -0.67134269 -0.66733467
-0.66332665 -0.65931864 -0.65531062 -0.65130261 -0.64729459 -0.64328657
-0.63927856 -0.63527054 -0.63126253 -0.62725451 -0.62324649 -0.61923848
-0.61523046 -0.61122244 -0.60721443 -0.60320641 -0.5991984  -0.59519038
-0.59118236 -0.58717435 -0.58316633 -0.57915832 -0.5751503  -0.57114228
-0.56713427 -0.56312625 -0.55911824 -0.55511022 -0.5511022  -0.54709419
-0.54308617 -0.53907816 -0.53507014 -0.53106212 -0.52705411 -0.52304609
-0.51903808 -0.51503006 -0.51102204 -0.50701403 -0.50300601 -0.498998
-0.49498998 -0.49098196 -0.48697395 -0.48296593 -0.47895792 -0.4749499
-0.47094188 -0.46693387 -0.46292585 -0.45891784 -0.45490982 -0.4509018
-0.44689379 -0.44288577 -0.43887776 -0.43486974 -0.43086172 -0.42685371
-0.42284569 -0.41883768 -0.41482966 -0.41082164 -0.40681363 -0.40280561
-0.3987976  -0.39478958 -0.39078156 -0.38677355 -0.38276553 -0.37875752
-0.3747495  -0.37074148 -0.36673347 -0.36272545 -0.35871743 -0.35470942
-0.3507014  -0.34669339 -0.34268537 -0.33867735 -0.33466934 -0.33066132
-0.32665331 -0.32264529 -0.31863727 -0.31462926 -0.31062124 -0.30661323
-0.30260521 -0.29859719 -0.29458918 -0.29058116 -0.28657315 -0.28256513
-0.27855711 -0.2745491  -0.27054108 -0.26653307 -0.26252505 -0.25851703
-0.25450902 -0.250501   -0.24649299 -0.24248497 -0.23847695 -0.23446894
-0.23046092 -0.22645291 -0.22244489 -0.21843687 -0.21442886 -0.21042084
-0.20641283 -0.20240481 -0.19839679 -0.19438878 -0.19038076 -0.18637275
-0.18236473 -0.17835671 -0.1743487  -0.17034068 -0.16633267 -0.16232465
-0.15831663 -0.15430862 -0.1503006  -0.14629259 -0.14228457 -0.13827655
-0.13426854 -0.13026052 -0.12625251 -0.12224449 -0.11823647 -0.11422846
-0.11022044 -0.10621242 -0.10220441 -0.09819639 -0.09418838 -0.09018036
-0.08617234 -0.08216433 -0.07815631 -0.0741483  -0.07014028 -0.06613226
-0.06212425 -0.05811623 -0.05410822 -0.0501002  -0.04609218 -0.04208417
-0.03807615 -0.03406814 -0.03006012 -0.0260521  -0.02204409 -0.01803607
-0.01402806 -0.01002004 -0.00601202 -0.00200401  0.00200401  0.00601202
 0.01002004  0.01402806  0.01803607  0.02204409  0.0260521   0.03006012
 0.03406814  0.03807615  0.04208417  0.04609218  0.0501002   0.05410822
 0.05811623  0.06212425  0.06613226  0.07014028  0.0741483   0.07815631
 0.08216433  0.08617234  0.09018036  0.09418838  0.09819639  0.10220441
 0.10621242  0.11022044  0.11422846  0.11823647  0.12224449  0.12625251
 0.13026052  0.13426854  0.13827655  0.14228457  0.14629259  0.1503006
 0.15430862  0.15831663  0.16232465  0.16633267  0.17034068  0.1743487
 0.17835671  0.18236473  0.18637275  0.19038076  0.19438878  0.19839679
 0.20240481  0.20641283  0.21042084  0.21442886  0.21843687  0.22244489
 0.22645291  0.23046092  0.23446894  0.23847695  0.24248497  0.24649299
 0.250501    0.25450902  0.25851703  0.26252505  0.26653307  0.27054108
 0.2745491   0.27855711  0.28256513  0.28657315  0.29058116  0.29458918
 0.29859719  0.30260521  0.30661323  0.31062124  0.31462926  0.31863727
 0.32264529  0.32665331  0.33066132  0.33466934  0.33867735  0.34268537
 0.34669339  0.3507014   0.35470942  0.35871743  0.36272545  0.36673347
 0.37074148  0.3747495   0.37875752  0.38276553  0.38677355  0.39078156
 0.39478958  0.3987976   0.40280561  0.40681363  0.41082164  0.41482966
 0.41883768  0.42284569  0.42685371  0.43086172  0.43486974  0.43887776
```

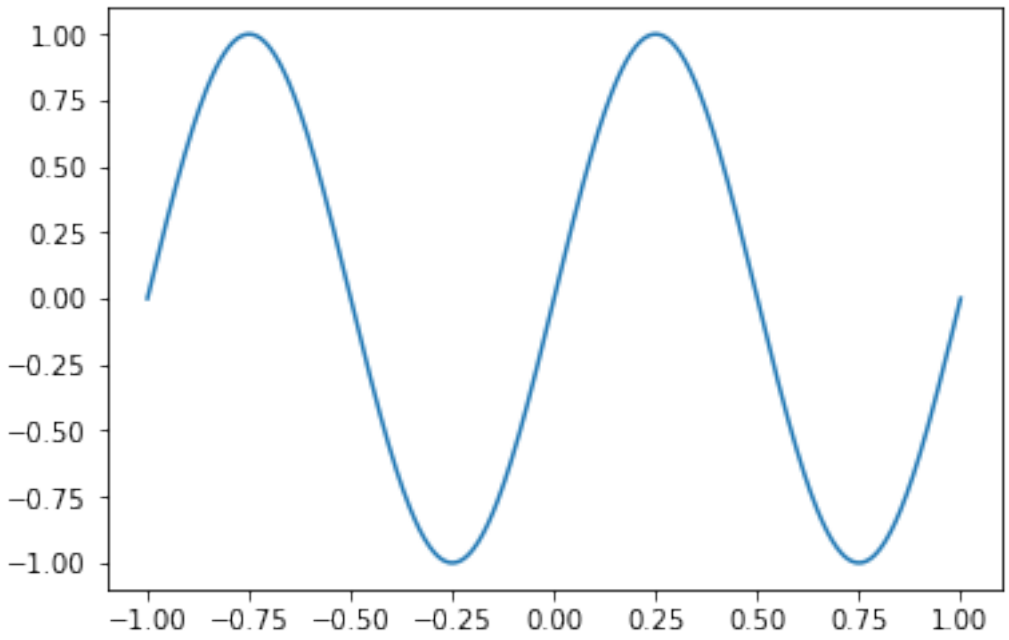
```
  0.44288577  0.44689379  0.4509018   0.45490982  0.45891784  0.46292585
  0.46693387  0.47094188  0.4749499   0.47895792  0.48296593  0.48697395
  0.49098196  0.49498998  0.498998    0.50300601  0.50701403  0.51102204
  0.51503006  0.51903808  0.52304609  0.52705411  0.53106212  0.53507014
  0.53907816  0.54308617  0.54709419  0.5511022   0.55511022  0.55911824
  0.56312625  0.56713427  0.57114228  0.5751503   0.57915832  0.58316633
  0.58717435  0.59118236  0.59519038  0.5991984   0.60320641  0.60721443
  0.61122244  0.61523046  0.61923848  0.62324649  0.62725451  0.63126253
  0.63527054  0.63927856  0.64328657  0.64729459  0.65130261  0.65531062
  0.65931864  0.66332665  0.66733467  0.67134269  0.6753507   0.67935872
  0.68336673  0.68737475  0.69138277  0.69539078  0.6993988   0.70340681
  0.70741483  0.71142285  0.71543086  0.71943888  0.72344689  0.72745491
  0.73146293  0.73547094  0.73947896  0.74348697  0.74749499  0.75150301
  0.75551102  0.75951904  0.76352705  0.76753507  0.77154309  0.7755511
  0.77955912  0.78356713  0.78757515  0.79158317  0.79559118  0.7995992
  0.80360721  0.80761523  0.81162325  0.81563126  0.81963928  0.82364729
  0.82765531  0.83166333  0.83567134  0.83967936  0.84368737  0.84769539
  0.85170341  0.85571142  0.85971944  0.86372745  0.86773547  0.87174349
  0.8757515   0.87975952  0.88376754  0.88777555  0.89178357  0.89579158
  0.8997996   0.90380762  0.90781563  0.91182365  0.91583166  0.91983968
  0.9238477   0.92785571  0.93186373  0.93587174  0.93987976  0.94388778
  0.94789579  0.95190381  0.95591182  0.95991984  0.96392786  0.96793587
  0.97194389  0.9759519   0.97995992  0.98396794  0.98797595  0.99198397
  0.99599198  1.        ]
```

6. What expression will yield the value of the 53th element of meshPoints? What is this value?

```
In [9]: print(meshPoints[52])

-0.7915831663326653
```

7. Produce a plot of a sinusoid on the interval [1, 1] using the command

```
In [10]: plt.plot(meshPoints,np.sin(2*math.pi*meshPoints))

Out[10]: [<matplotlib.lines.Line2D at 0x619d35f28>]
```




```
In [11]: plt.savefig('meshPoints.jpg')
```

```
<Figure size 432x288 with 0 Axes>
```

1.2.2 Resuelva los ejercicios de las secciones 4.1, 5.1, 6.1, 7.4 y 8.5 de este documento.

1.3 Scientific Plotting with Matplotlib

1.3.1 4. Simple Plots

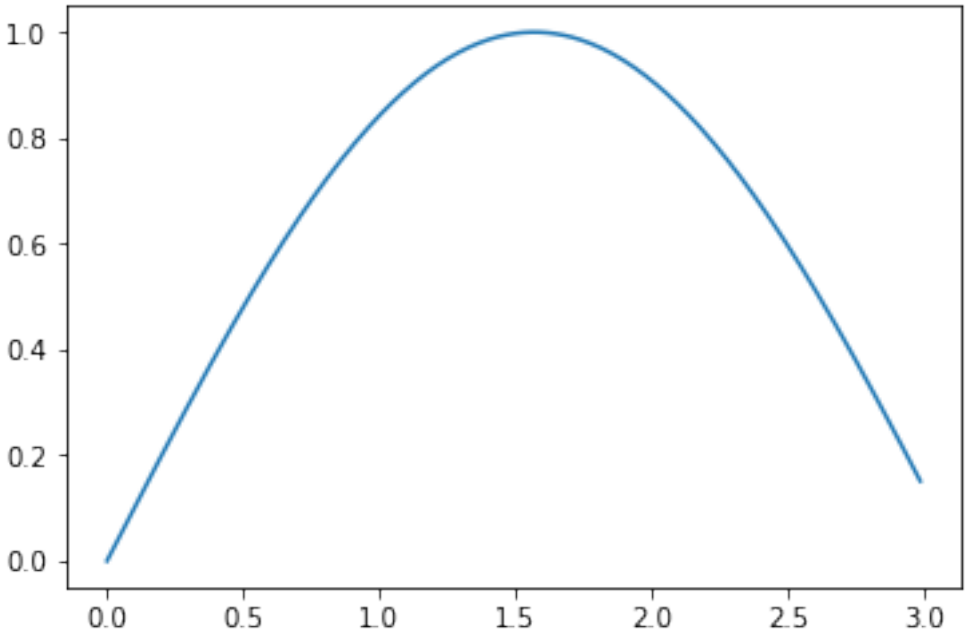
1.3.2 4.1. Exercises

1. Plot a simple graph of a sinus function in the range 0 to 3 with a step size of 0.01.

```
In [12]: import matplotlib.pyplot as plt
         %matplotlib inline
```

```
In [13]: r = np.arange(0,3,0.01)
         plt.plot(r, np.sin(r))
```

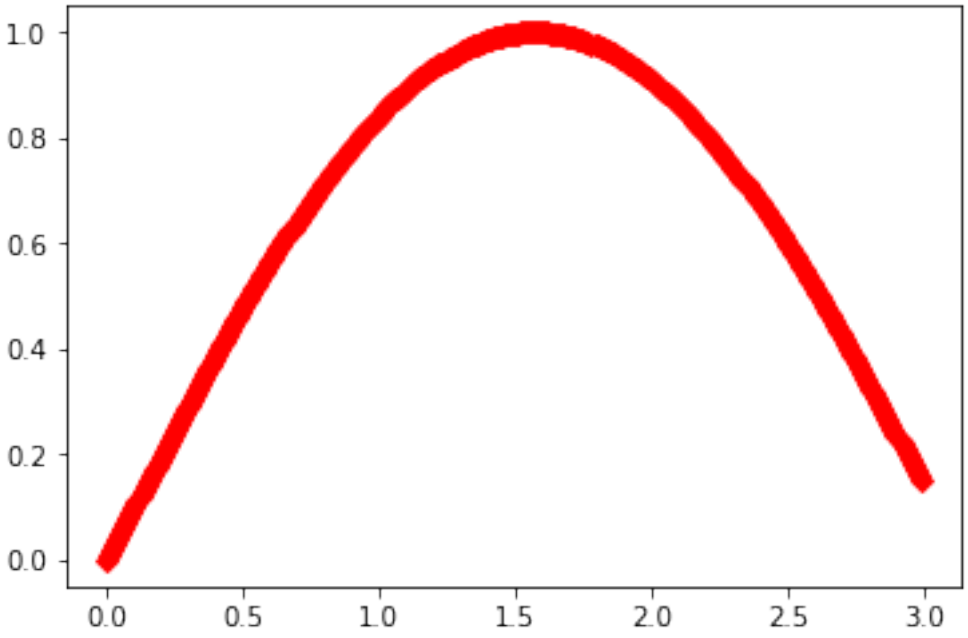
```
Out[13]: [<matplotlib.lines.Line2D at 0x619ede0b8>]
```



2. Make the line red. Add diamond-shaped markers with size of 5.

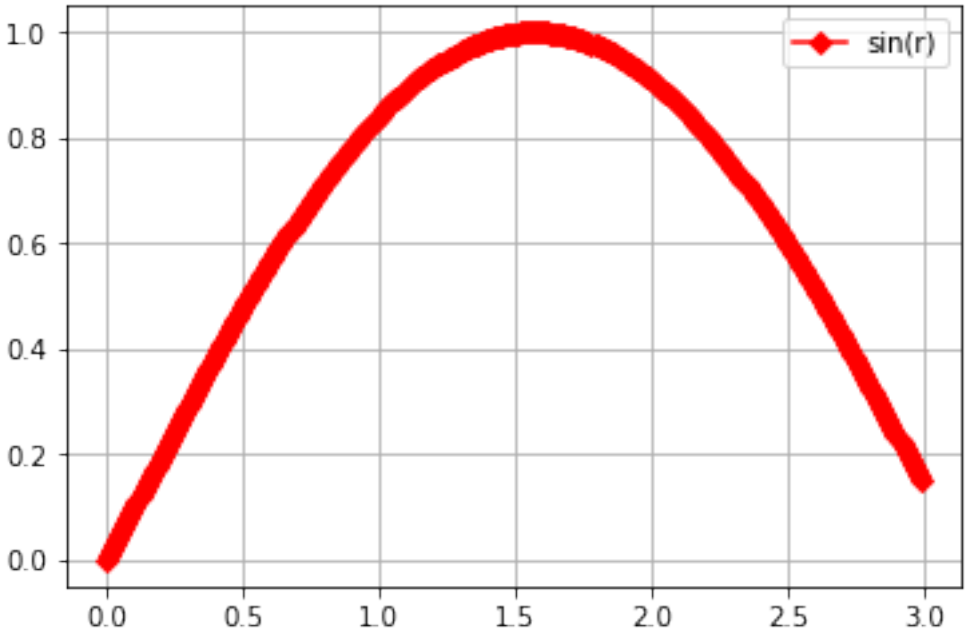
```
In [14]: plt.plot(r, np.sin(r), marker='D', markersize=5, color='r')
```

```
Out[14]: [<matplotlib.lines.Line2D at 0x619e44828>]
```



3. Add a legend and a grid to the plot.

```
In [15]: plt.plot(r, np.sin(r), marker='D', markersize=5, color='r', label="sin(r)")
         plt.legend();
         plt.grid(True)
```

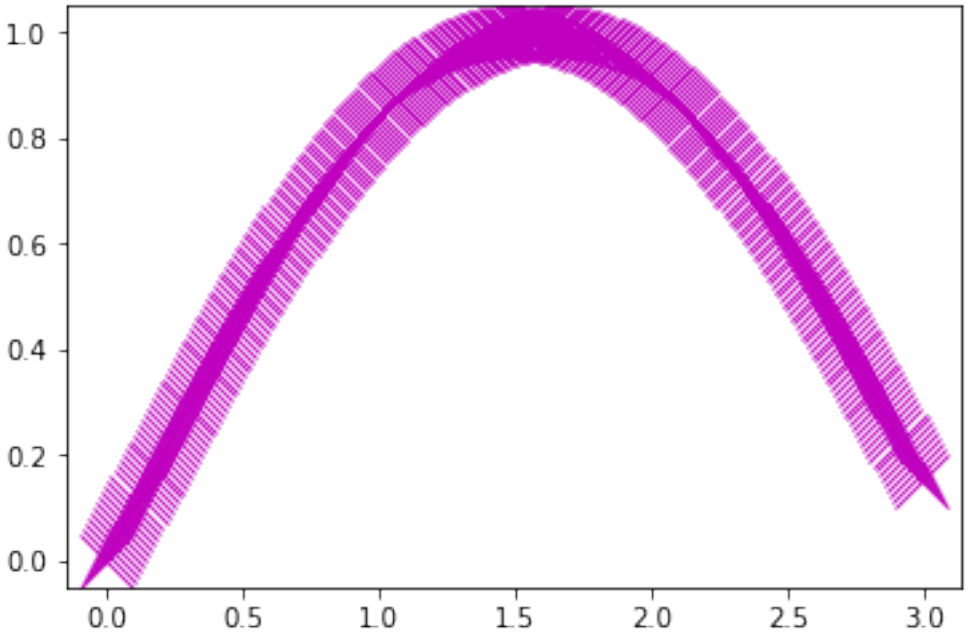


1.3.3 5. Properties

1.3.4 5.1. Exercise

1. Apply different line styles to a plot. Change line color and thickness as well as the size and the kind of the marker. Experiment with different styles.

```
In [16]: plt.plot(r, np.sin(r), marker='x', markersize=20, color='m',linewidth=1)

Out[16]: [<matplotlib.lines.Line2D at 0x61a220828>]
```



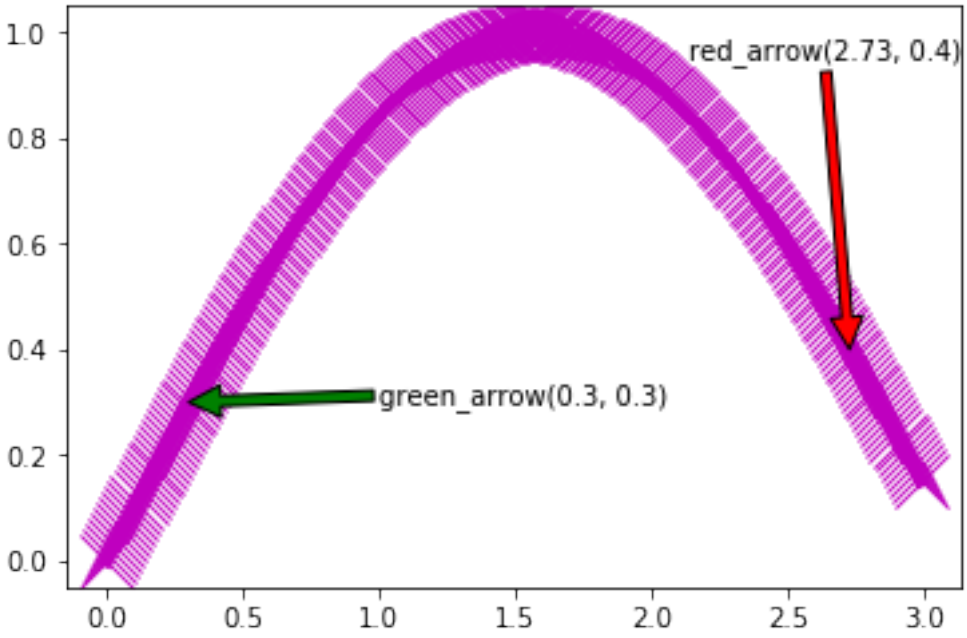
1.3.5 6. Text

1.3.6 6.1. Exercise

1. Annotate a line at two places with text. Use green and red arrows and align it according to figure points and data.

```
In [17]: plt.plot(r, np.sin(r), marker='x', markersize=20, color='m',linewidth=5)
         plt.annotate('green_arrow(0.3, 0.3)', xy=(0.3, 0.3), xytext=(1.0,0.3),arrowprops=dict
         plt.annotate('red_arrow(2.73, 0.4)', xy=(2.73, 0.4), xytext=(2.14,0.95),arrowprops=di
```

```
Out[17]: Text(2.14, 0.95, 'red_arrow(2.73, 0.4)')
```

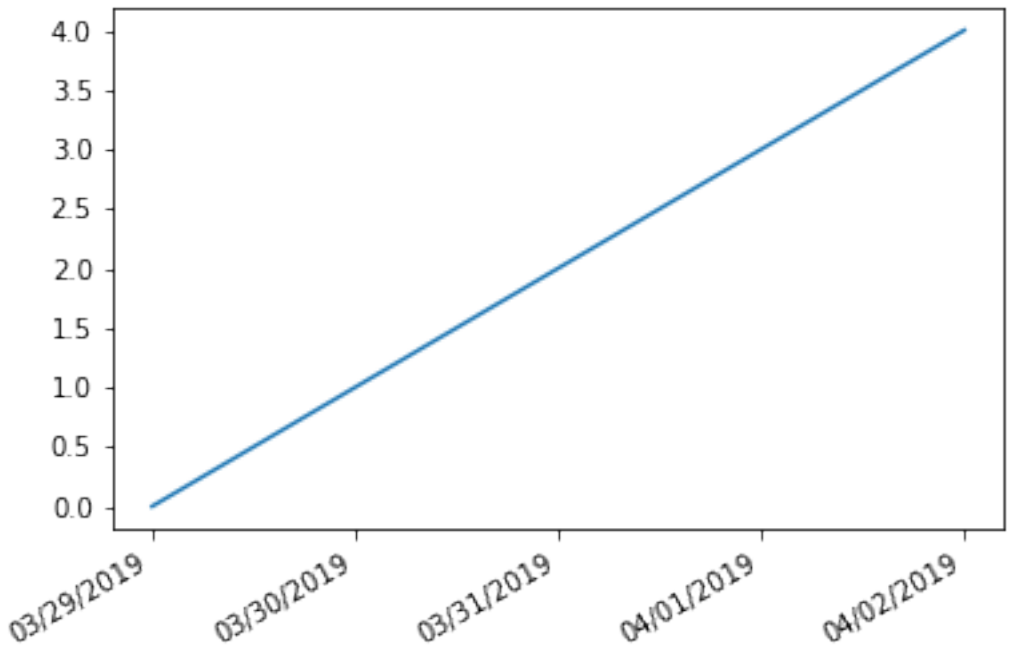


1.3.7 7. Ticks

1.3.8 7.4. Exercises

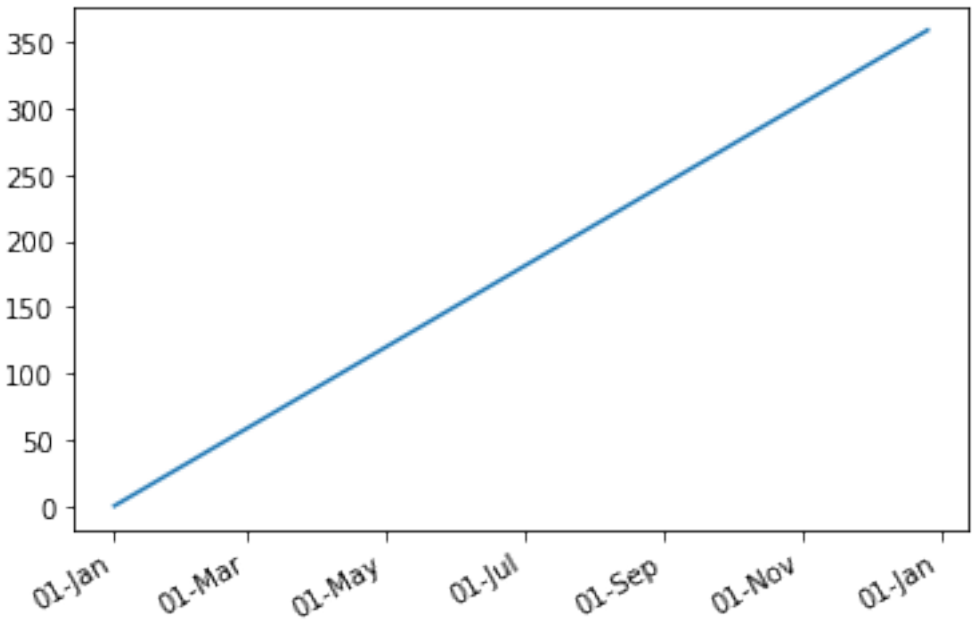
1. Plot a graph with dates for one year with daily values at the x axis using the built-in module datetime.

```
In [18]: import datetime as dt
         import matplotlib.dates as mdates

In [19]: dates = ['03/29/2019','03/30/2019','03/31/2019','04/01/2019','04/02/2019']
         x = [dt.datetime.strptime(d,'%m/%d/%Y').date() for d in dates]
         y = range(len(x))

In [20]: plt.gca().xaxis.set_major_formatter(mdates.DateFormatter('%m/%d/%Y'))
         plt.gca().xaxis.set_major_locator(mdates.DayLocator())
         plt.plot(x,y)
         plt.gcf().autofmt_xdate()
```



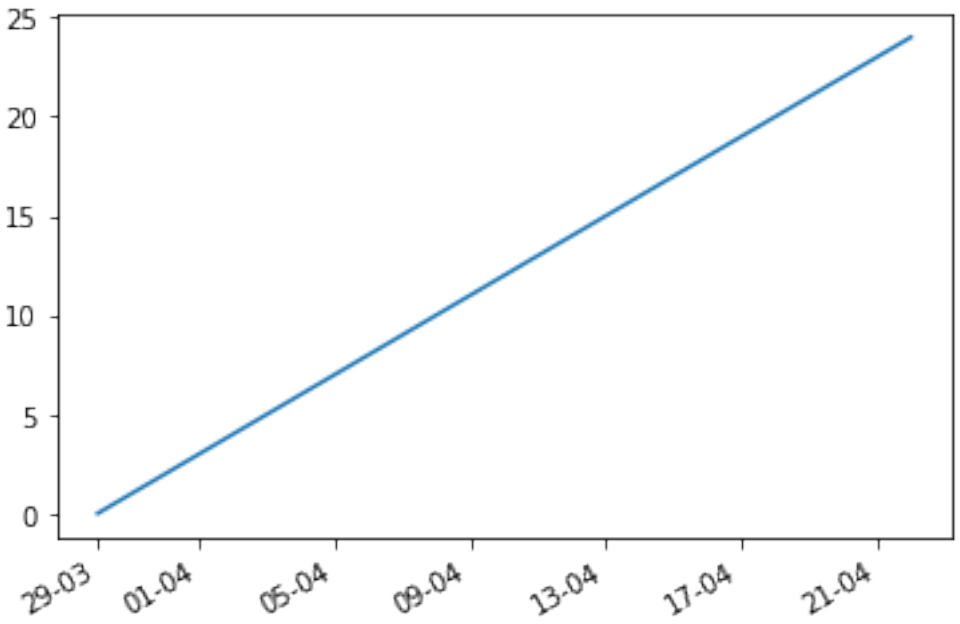
2. Format the dates in such a way that only the first day of the month is shown.

```
In [21]: import pandas as pd
         import matplotlib.dates as mdates

In [22]: times = pd.date_range('2019-01-01', periods=360)
         fig, ax = plt.subplots(1)
         fig.autofmt_xdate()
         plt.plot(times, range(times.size))
         xfmt = mdates.DateFormatter('%d-%b')
         ax.xaxis.set_major_formatter(xfmt)
         plt.show()
```

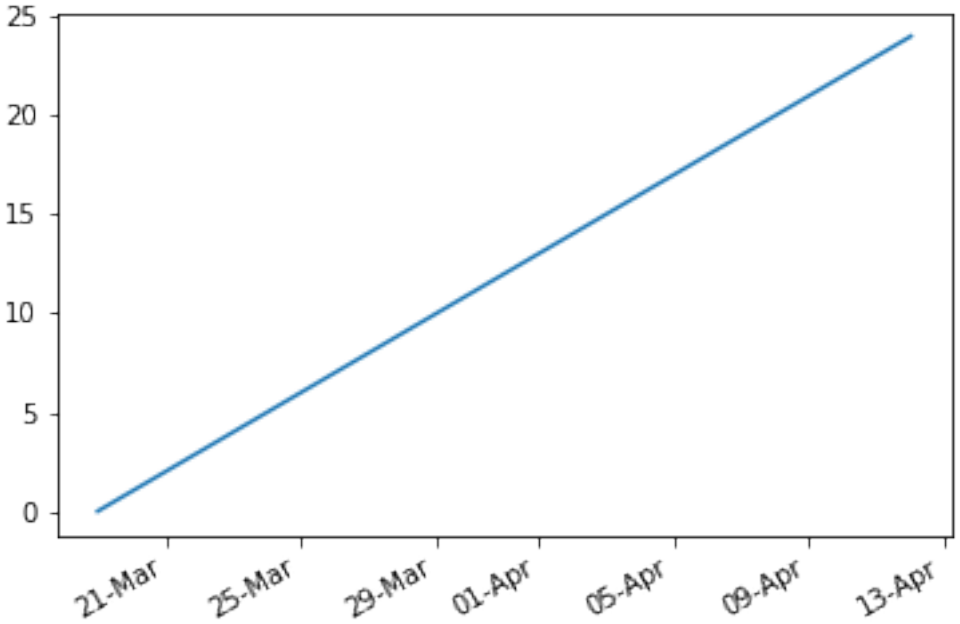


3. Display the dates with and without the year. Show the month as number and as first three letters of the month name

```
In [23]: times = pd.date_range('2019-03-29', periods=25)

         fig, ax = plt.subplots(1)
         fig.autofmt_xdate()
         plt.plot(times, range(times.size))
         xfmt = mdates.DateFormatter('%d-%m')
         ax.xaxis.set_major_formatter(xfmt)
         plt.show()
```



In [24]:
```
times = pd.date_range('2019-03-19', periods=25)
fig, ax = plt.subplots(1)
fig.autofmt_xdate()
plt.plot(times, range(times.size))
xfmt = mdates.DateFormatter('%d-%b')
ax.xaxis.set_major_formatter(xfmt)
plt.show()
```



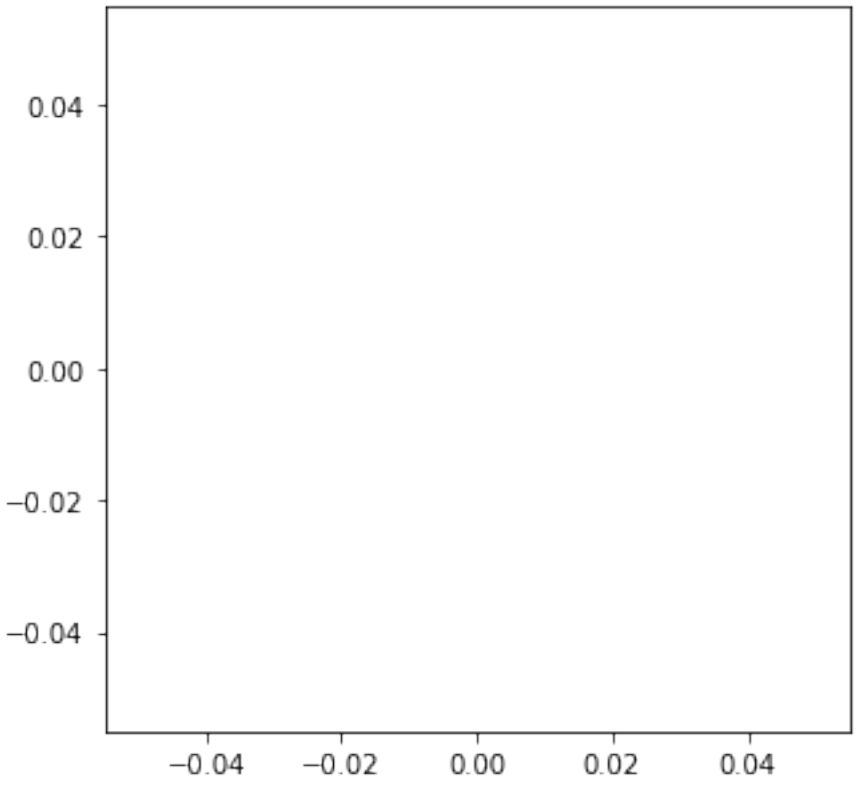
1.3.9 8. Figures, Subplots, and Axes

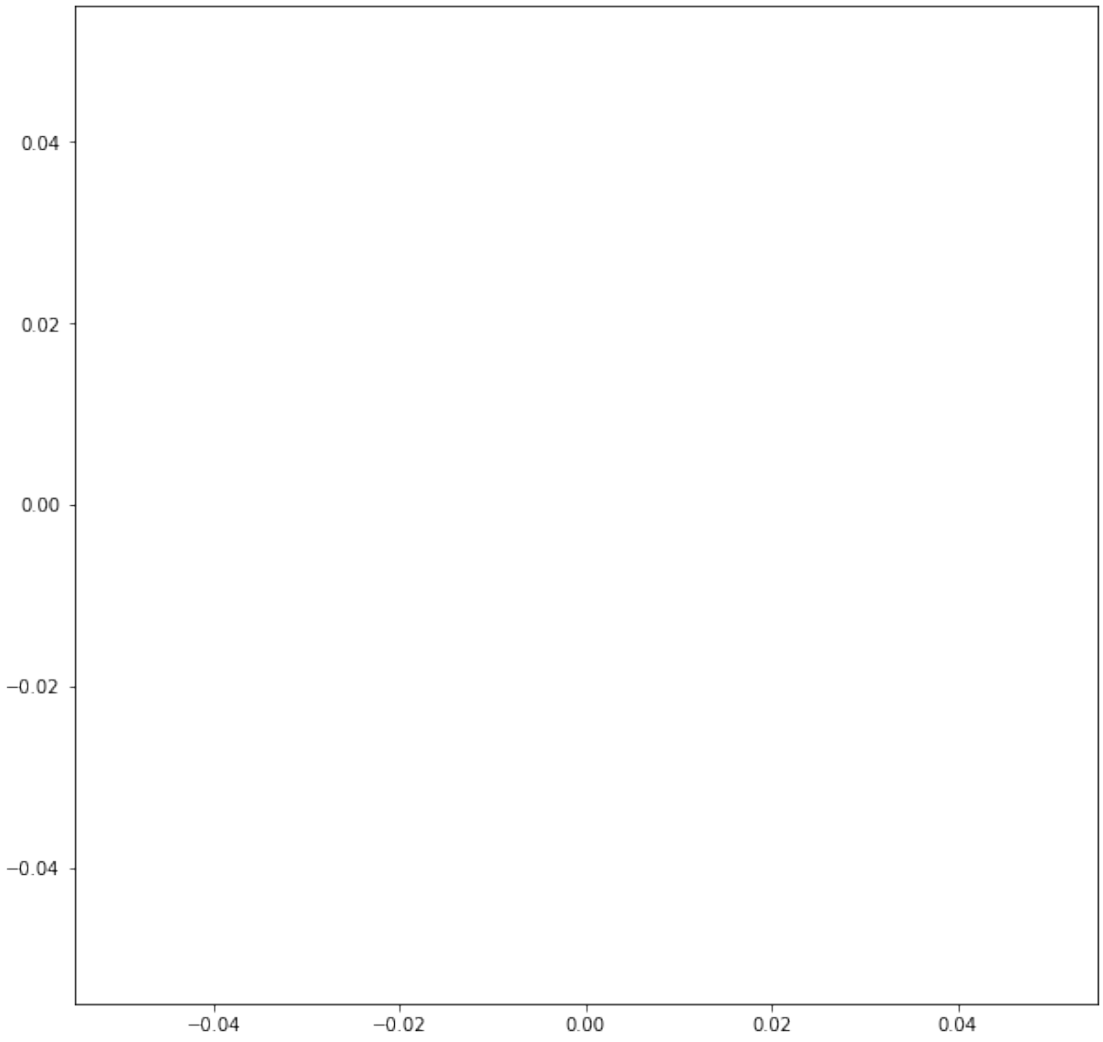
1.3.10 8.5. Exercises

1. Draw two figures, one 5 by 5, one 10 by 10 inches.

```
In [25]: plt.figure(figsize=(5,5))
         plt.plot()
         plt.figure(figsize=(10,10))
         plt.plot()
```

```
Out[25]: []
```



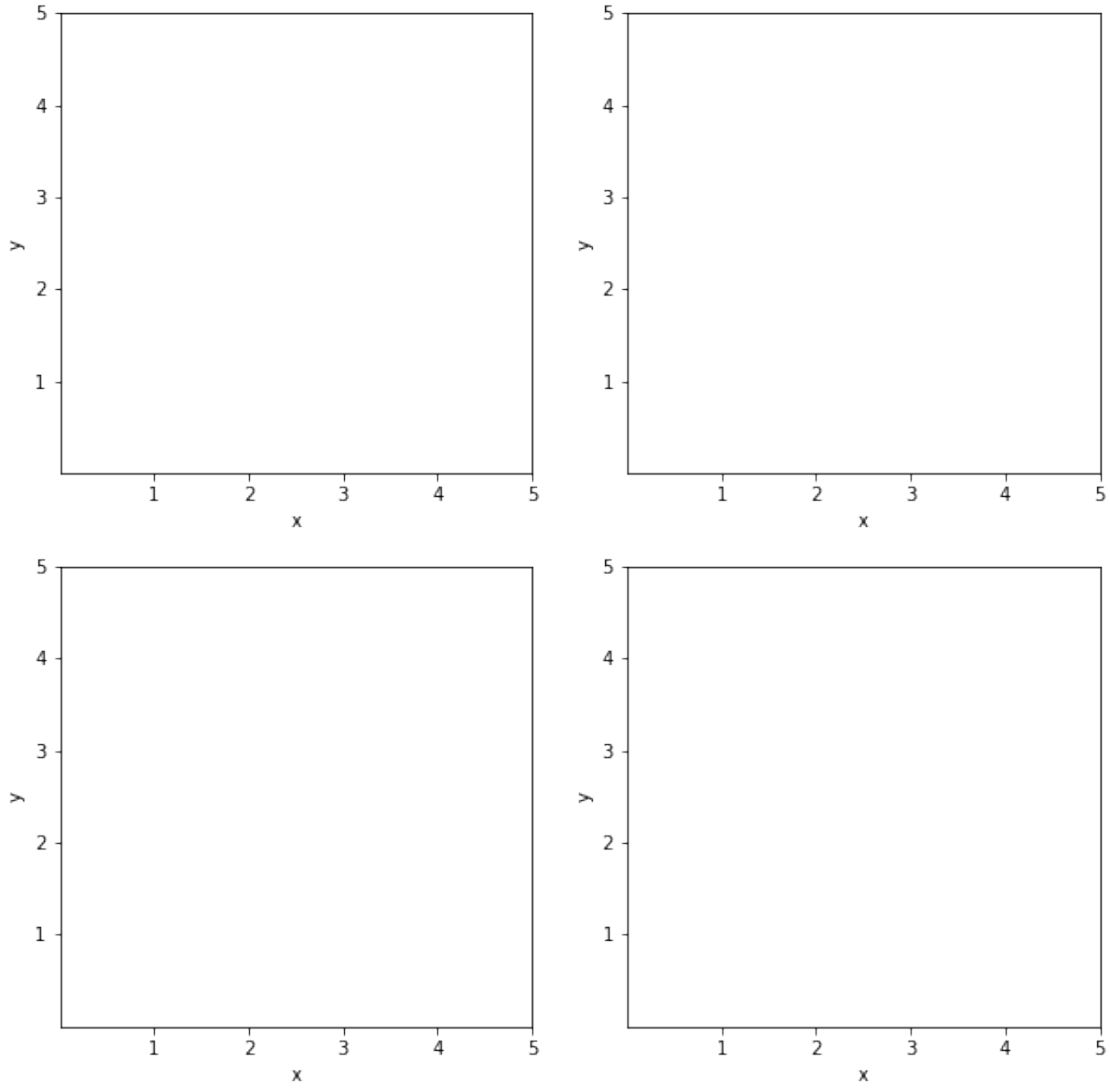


2. Add four subplots to one figure. Add labels and ticks only to the outermost axes.

```
In [26]: plt.figure(figsize=(10,10))
         plt.subplot(221)
         plt.xlabel("x")
         plt.ylabel("y")
         plt.xticks([1,2,3,4,5])
         plt.yticks([1,2,3,4,5])
         plt.subplot(222)
         plt.xlabel("x")
         plt.ylabel("y")
         plt.xticks([1,2,3,4,5])
         plt.yticks([1,2,3,4,5])
         plt.subplot(223)
         plt.xlabel("x")
```

```
plt.ylabel("y")
plt.xticks([1,2,3,4,5])
plt.yticks([1,2,3,4,5])
plt.subplot(224)
plt.xlabel("x")
plt.ylabel("y")
plt.xticks([1,2,3,4,5])
plt.yticks([1,2,3,4,5])
```

```
Out[26]: ([<matplotlib.axis.YTick at 0x61bf39c88>,
  <matplotlib.axis.YTick at 0x61bf39550>,
  <matplotlib.axis.YTick at 0x61bf22be0>,
  <matplotlib.axis.YTick at 0x61c118d68>,
  <matplotlib.axis.YTick at 0x61c118e48>],
 <a list of 5 Text yticklabel objects>)
```



3. Place a small plot in one bigger plot.

```
In [27]: fig = plt.figure()
         ax1 = fig.add_axes([0.5, 0.5, 0.5, 0.5])
         ax2 = fig.add_axes([0.72, 0.72, 0.16, 0.16])
```