[Python] 시각화 사용법 - matplotlib을 통한 line plot 그리기(lim,ticks 등등) — 나무늘보의 개발 블로그

노트북:	첫 번째 노트북
만든 날짜:	2020-11-08 오후 9:20
URL:	https://continuous-development.tistory.com/143?category=736681

Python

[Python] 시각화 사용법 - matplotlib을 통한 line plot 그리기(lim,ticks 등등)

2020. 10. 19. 17:35 수정 삭제 공개

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

기본적인 함수를 넣어준다. 그 외에 추가적으로 그래프에 한글이 깨지는 경우가 많아 함수를 하나 추가해준다.

In [3]:
```
%matplotlib inline
import platform

from matplotlib import font_manager, rc
# plt.rcParams['axes.unicode_minus'] = False

if platform.system() == 'Darwin':
    rc('font', family='AppleGothic')
elif platform.system() == 'Windows':
    path = "c:/Windows/Fonts/malgun.ttf"
    font_name = font_manager.FontProperties(fname=path).get_name()
    rc('font', family=font_name)
else:
    print('Unknown system... sorry~~~~')
```

line plot

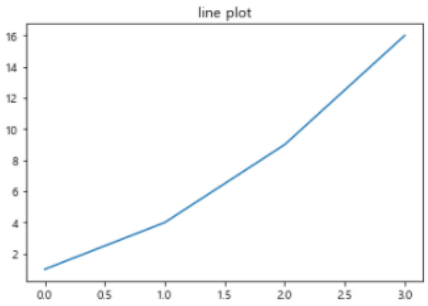
데이터의 시간 순서등에 따라 어떻게 변화하는지 보여주는 그래프

-pyplot (기본적인 그래프 제공)

```
plt.title('line plot')
plt.plot([1,4,9,16])
plt.show()
```

plot을 통해 해당 값에 대한 선을 그린다.
title은 그래프의 제목을 뜻한다.
plt.show()는 그래프를 출력해준다.

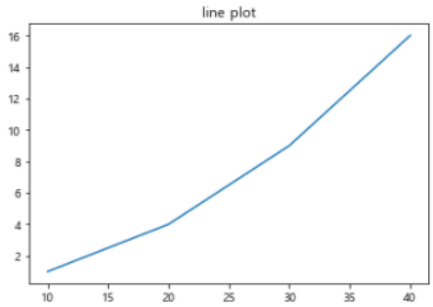
In [5]:
```
plt.title('line plot')
plt.plot([1,4,9,16])
plt.show()
```



-x축 y축 지정

```
plot([ x 축],[y 축])
```

In [6]:
```
plt.title('line plot')
plt.plot([10,20,30,40] , [1,4,9,16]) # 앞에 x 축 , 뒤가 y 축
plt.show()
```

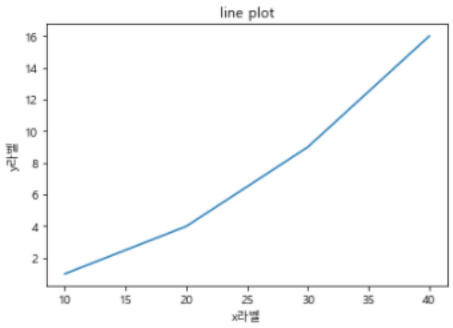


-label 사용

```
plt.xlabel()
ply.ylabel()
```

x.label , ylabel 을 통해 x 축 label y축 label을 지정해준다.

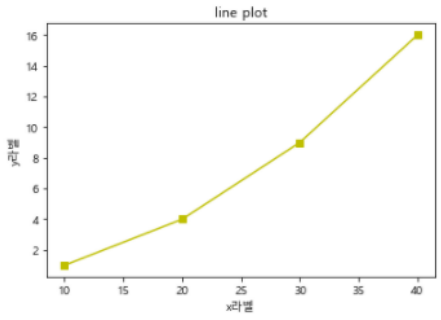
```
In [7]: plt.title('line plot')
        plt.xlabel('x라벨')
        plt.ylabel('y라벨')
        plt.plot([10,20,30,40] , [1,4,9,16]) # 앞에 x 축 , 뒤가 y 축
        plt.show()
```



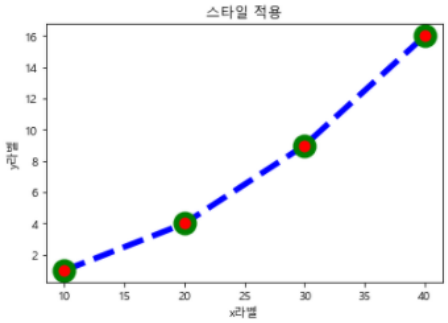
-옵션 사용

```
- option(color, marker, line style)
- c(color) / lw(라인굵기) / ls(라인 스타일) / mec(마커선 색깔)/ mew (마커선 굵기) / mfc(마커내부색깔) / ms(마커 스타일)
```

```
In [11]: plt.title('line plot')
         plt.xlabel('x라벨')
         plt.ylabel('y라벨')
         plt.plot([10,20,30,40] , [1,4,9,16], 'ys-') # 앞에 x 축 , 뒤가 y 축
         plt.show()
```

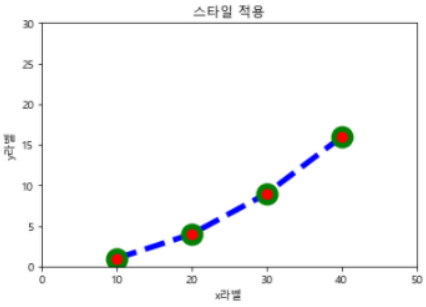


In [13]:
```
plt.title('스타일 적용')
plt.xlabel('x라벨')
plt.ylabel('y라벨')
plt.plot([10,20,30,40] , [1,4,9,16], c='b',lw=5,ls='--',marker='o',ms=15,mec='g',mew=5,mfc='r') # 앞에 x 축 , 뒤가 y 축
plt.show()
```



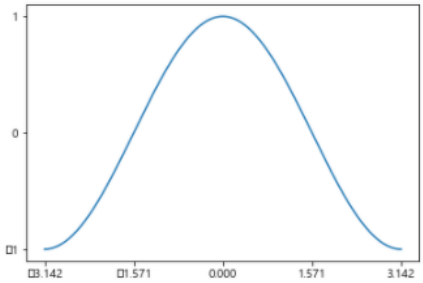
-xlim, ylim : 구간설정

In [15]:
```
plt.title('스타일 적용')
plt.xlabel('x라벨')
plt.ylabel('y라벨')
plt.plot([10,20,30,40] , [1,4,9,16], c='b',lw=5,ls='--',marker='o',ms=15,mec='g',mew=5,mfc='r') # 앞에 x 축 , 뒤가 y 축
plt.xlim(0,50) # lim으로 구간을 만들수 있다.
plt.ylim(0,30)
plt.show()
```

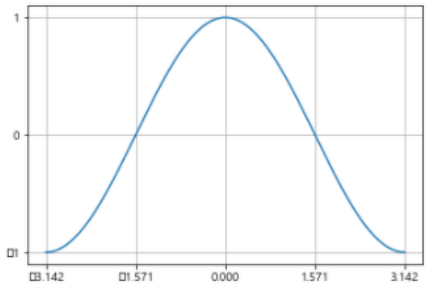


- tick(xticks, yticks) : 그래프의 축에 간격을 만든다.

In [18]:
```
X = np.linspace(-np.pi, np.pi,256)
Y = np.cos(X)
plt.plot(X,Y)
plt.xticks([-np.pi,-np.pi/2,0 ,np.pi/2,np.pi])
plt.yticks([-1,0,1])
plt.show()
```

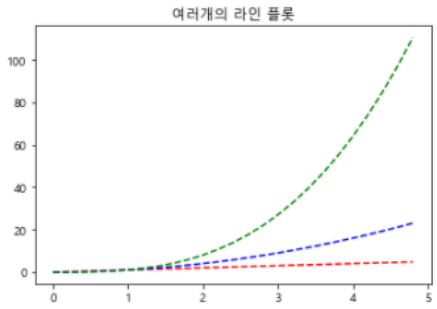


In [20]:
```
X = np.linspace(-np.pi, np.pi,256)
Y = np.cos(X)
plt.plot(X,Y)
plt.xticks([-np.pi,-np.pi/2,0 ,np.pi/2,np.pi])
plt.yticks([-1,0,1])
plt.grid(True) # tick에 따라 격자무늬 생성
plt.show()
```



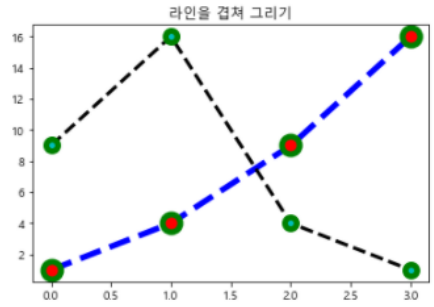
-여러 개의 line plot 그리기

In [25]:
```
data  = np.arange(0. , 5.0, 0.2)
data
plt.title('여러개의 라인 플롯')
plt.plot(data,data,'r--',data,data**2, 'b--',data,data**3, 'g--')
plt.show()
```



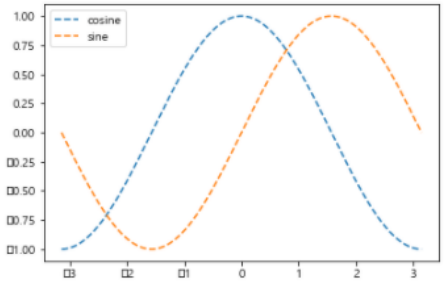
-라인 겹쳐 그리기

In [26]:
```
plt.title('라인을 겹쳐 그리기')
plt.plot( [1,4,9,16], c='b',lw=5,ls='--',marker='o',ms=15,mec='g',mew=5,mfc='r') # 앞에 x 축 , 뒤가 y 축
plt.plot( [9,16,4,1], c='k',lw=3,ls='--',marker='o',ms=10,mec='g',mew=5,mfc='c') # 앞에 x 축 , 뒤가 y 축
plt.show()
```



-legend 위치

In [34]:
```
X = np.linspace(-np.pi, np.pi, 256)
Y_C, Y_S = np.cos(X), np.sin(X)
plt.plot(X,Y_C, ls='--', label='cosine')
plt.plot(X,Y_S, ls='--', label='sine')
plt.legend(loc=0)
plt.show()
```



matplotlib

- matplotlib 그림을 그리는 객체 , Figure, Axes, Axis 객체를 포함하고 있다.
- Figure -> 그래프를 그리는 바탕을 뜻한다.
- Axes -> plot

- Axis -> 축

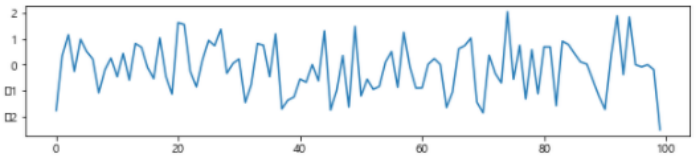
```
In [36]: np.random.seed(100)
         plt.figure(figsize=(10,2))
         plt
         # 0개의 plot과 720, 144의 사이즈를 가지고있다.
```

```
Out[36]: <module 'matplotlib.pyplot' from 'C:₩₩Users₩₩hwang in beom₩₩Anaconda3₩₩lib₩₩site-packages₩₩matplotlib₩₩pyplot.py'>

         <Figure size 720x144 with 0 Axes>
```

figsize로 그래프를 그릴 바탕의 크기를 정한다.

```
In [37]: np.random.seed(100)
         plt.figure(figsize=(10,2))
         plt.plot(np.random.randn(100))
         plt.show()
         # 0개의 plot과 720, 144의 사이즈를 가지고있다.
```



지금은 하나의 figure에 하나의 그래프만 나왔지만 이것을 두 개로 나오는 것을 해보기로 하자

subplot

하나의 figure에 두 가지 plot으로 그래프를 그린다.

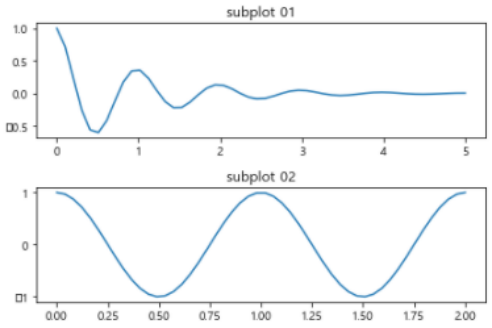
- subplot(2,1,1) # 하나의 figure에 두가지 plot으로 그린다.
- subplot(2,1,2)

In [46]:
```
X1 = np.linspace(0.0,5.0)
Y1 = np.cos(2*np.pi*X1)*np.exp(-X1)
print(X1)
print(Y1)
X2= np.linspace(0.0,2.0)
Y2 = np.cos(2*np.pi*X2)
print(X2)
print(Y2)
```

```
[0.         0.10204082 0.20408163 0.30612245 0.40816327 0.51020408
 0.6122449  0.71428571 0.81632653 0.91836735 1.02040816 1.12244898
 1.2244898  1.32653061 1.42857143 1.53061224 1.63265306 1.73469388
 1.83673469 1.93877551 2.04081633 2.14285714 2.24489796 2.34693878
 2.44897959 2.55102041 2.65306122 2.75510204 2.85714286 2.95918367
 3.06122449 3.16326531 3.26530612 3.36734694 3.46938776 3.57142857
 3.67346939 3.7755102  3.87755102 3.97959184 4.08163265 4.18367347
 4.28571429 4.3877551  4.48979592 4.59183673 4.69387755 4.79591837
 4.89795918 5.        ]
[ 1.          0.72367065  0.2320026  -0.25429107 -0.55721991 -0.59913951
 -0.41280458 -0.10893327  0.17893551  0.34780448  0.35748852  0.23380958
  0.04690762 -0.12275601 -0.21591812 -0.21241248 -0.13137462 -0.01694367
  0.08259904  0.1333648   0.12567347  0.0731473   0.00339549 -0.05472957
 -0.08198104 -0.07402828 -0.04029719  0.00203856  0.03580865  0.05016511
  0.04340781  0.02192149 -0.00366666 -0.0231816  -0.03056191 -0.02533133
 -0.01174303  0.00365889  0.01487088  0.0185398   0.01470776  0.00616987
 -0.00306273 -0.00946371 -0.01119988 -0.00849338 -0.00316048  0.00235117
  0.00597998  0.00673795]
[0.         0.04081633 0.08163265 0.12244898 0.16326531 0.20408163
 0.24489796 0.28571429 0.32653061 0.36734694 0.40816327 0.44897959
 0.48979592 0.53061224 0.57142857 0.6122449  0.65306122 0.69387755
 0.73469388 0.7755102  0.81632653 0.85714286 0.89795918 0.93877551
 0.97959184 1.02040816 1.06122449 1.10204082 1.14285714 1.18367347
 1.2244898  1.26530612 1.30612245 1.34693878 1.3877551  1.42857143
 1.46938776 1.51020408 1.55102041 1.59183673 1.63265306 1.67346939
 1.71428571 1.75510204 1.79591837 1.83673469 1.87755102 1.91836735
 1.95918367 2.        ]
[ 1.          0.96729486  0.8713187   0.71834935  0.51839257  0.28452759
  0.03205158 -0.22252093 -0.46253829 -0.67230089 -0.8380881  -0.94905575
 -0.99794539 -0.98155916 -0.90096887 -0.76144596 -0.57211666 -0.34536505
 -0.09602303  0.1595999   0.40478334  0.6234898   0.80141362  0.92691676
  0.99179001  0.99179001  0.92691676  0.80141362  0.6234898   0.40478334
  0.1595999  -0.09602303 -0.34536505 -0.57211666 -0.76144596 -0.90096887
 -0.98155916 -0.99794539 -0.94905575 -0.8380881  -0.67230089 -0.46253829
 -0.22252093  0.03205158  0.28452759  0.51839257  0.71834935  0.8713187
  0.96729486  1.        ]
```

그래프에 들어갈 값을 만든다.

In [48]:
```
axes01 = plt.subplot(2,1,1)
plt.plot(X1,Y1)
plt.title('subplot 01')
axes02 = plt.subplot(2,1,2)
plt.plot(X2,Y2)
plt.title('subplot 02')
plt.tight_layout() # 자동으로 간격을 맞춰주는 역할
plt.show()
```



subplot(x, y , z)

x = nrows, subplot의 행의 수를 나타낸다.

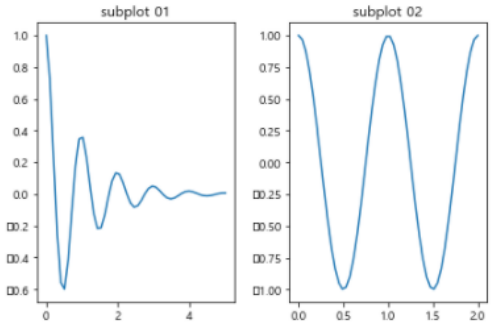
y = ncols, subplot의 열의 수를 나타낸다.

z = index, 인덱스를 나타낸다. 1부터 시작하며 순서는 위부토 오른쪽으로 위쪽에서 아래쪽 방향으로 설정된다.

이런 식으로 subplot을 보면 첫 번째 값은 2행을 가지며 열의수와 인덱스는 1을 나타낸다. 2행 1열 첫번째 인덱스에 그래프를 그린다고 생각하면 된다.

밑에 값은 2행 1열에 2번째 인덱스에 그래프를 그렸다.

In [49]:
```
axes01 = plt.subplot(1,2,1)
plt.plot(X1,Y1)
plt.title('subplot 01')
axes02 = plt.subplot(1,2,2)
plt.plot(X2,Y2)
plt.title('subplot 02')
plt.tight_layout() # 자동으로 간격을 맞춰주는 역할
plt.show()
```



이렇게 subplot의 값이 바뀜에 따라 그려지는 위치가 달라진다.

'**Python**' 카테고리의 다른 글□

[Python] matplotlib을 통한 bar plot 그리기□

[Python] matplotlib 한글 폰트 깨짐 현상 , 마이너스 기호(폰트)가 깨지는 현상 해결방법□

[Python] 시각화 사용법 - matplotlib을 통한 line plot 그리기(lim,ticks 등등)□

[Python] Pandas 사용법 - 피벗 테이블 생성(pivot,pivot_table)□

[Python] Pandas 사용법 - 그룹화 및 그룹 함수 (groupby, qcut, cut, transfrom)□

[Python] Pandas 사용법 - 두가지의 DataFrame 합치기 (merge, join)□

python line plot

python matplotlib

파이썬 line 그래프 그리기

파이썬 matplotlib

나무늘보스

혼자 끄적끄적하는 블로그 입니다.