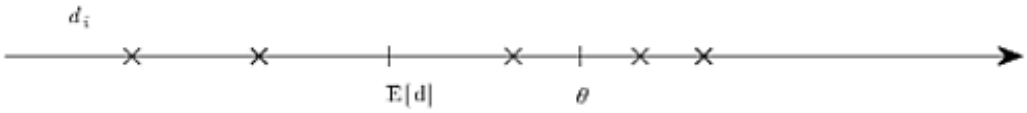
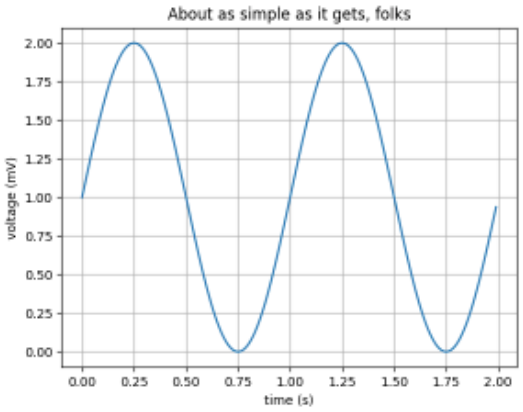
Plotting using Python:

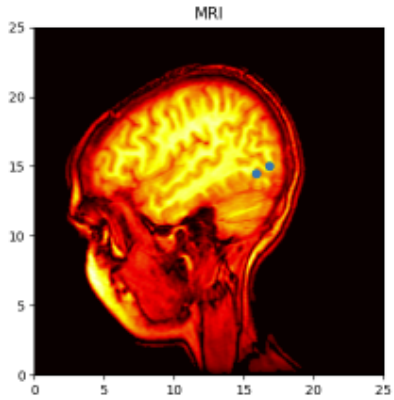
1. Plot dimension

- 1D
  x=[1,2,3,4,5,6,7]

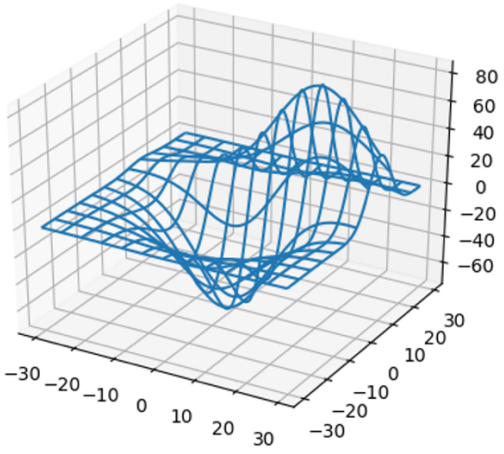


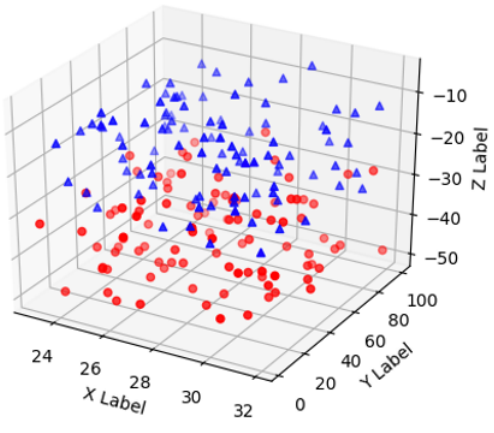
- 2D
  x=[-2,-1,0,1,2,3,4,5,6]
  y=[2,4,1,2,3,4,5,6,7,1]



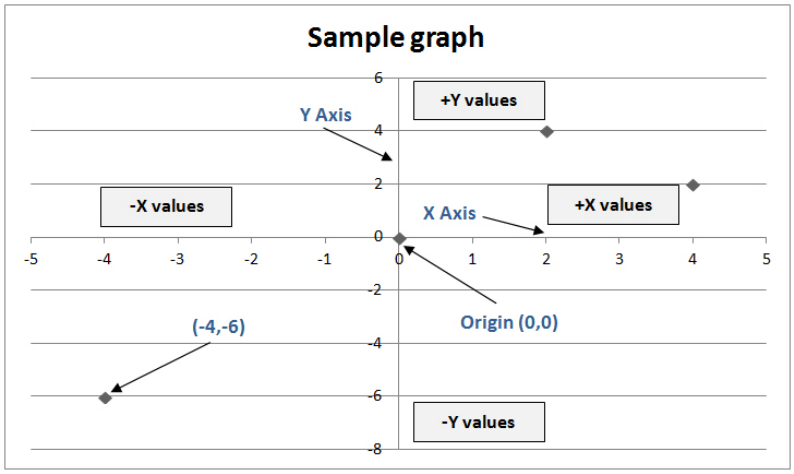


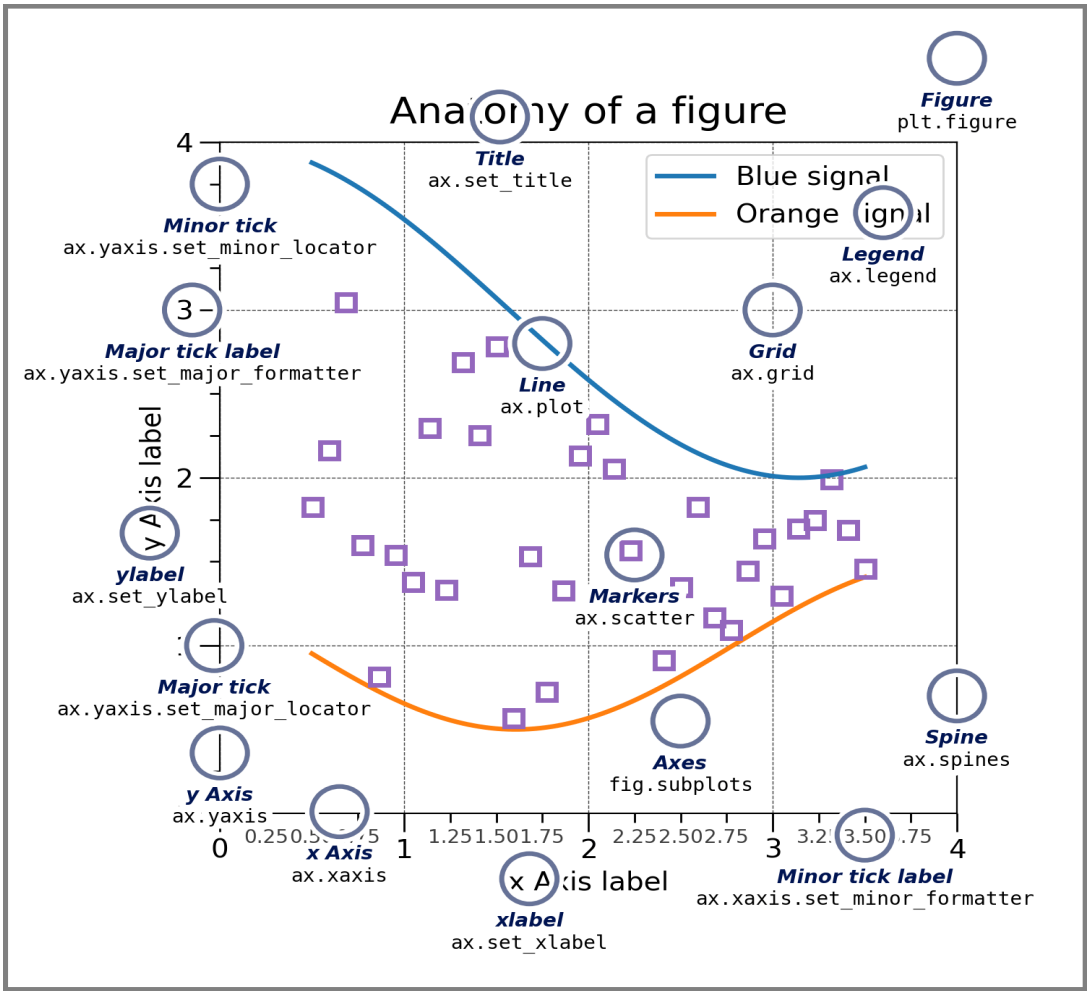
- 3D





2. Plotting Basics





3. Create (obtain) Data:

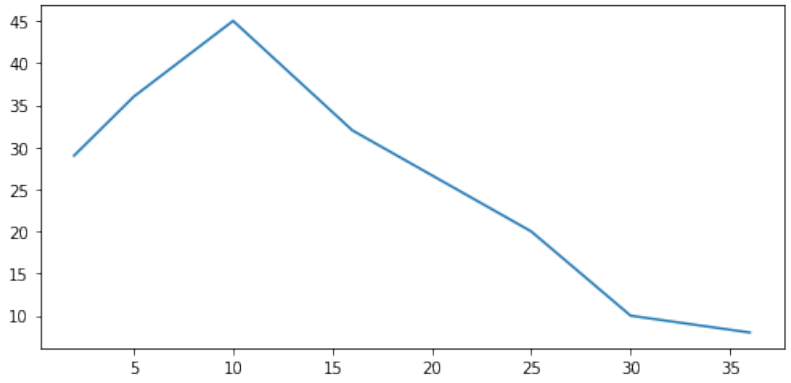
Depth (m)	Number of bubbles
2	29
5	36
10	45
16	32
25	20
30	10
36	8

```
D=[2,5,10,16,25,30,36]  # independent variable (x)
B=[29,36,45,32,20,10,8] # dependent variable  (y)


Data=[[2,5,10,16,25,30,36],[29,36,45,32,20,10,8]]

D = Data[0]
B = Data[1]
```

4. Plotting (Basic)

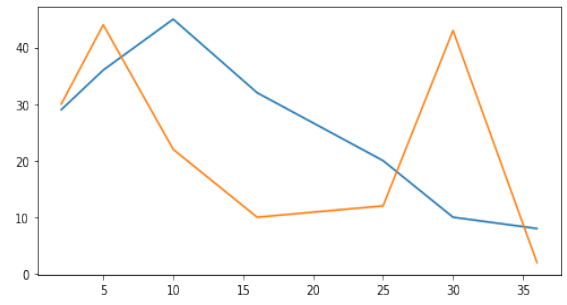
Plot Basic Data

```
1.  import matplotlib.pyplot as plt # call matplotlib pyplot library
2.
3.  plt.plot(D,B)
4.  plt.show()
```

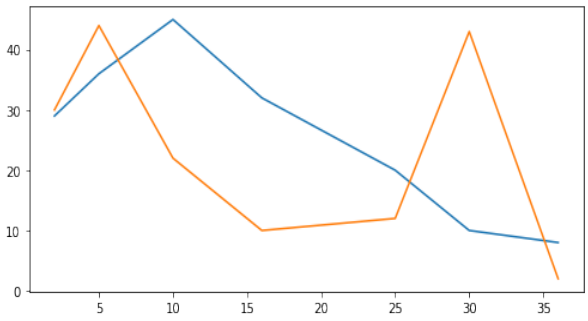


Multiple plots in the same graph.

```
1.  D=[2,5,10,16,25,30,36]
2.  B=[29,36,45,32,20,10,8]
3.  T=[30,44,22,10,12,43,2]
4.
5.  plt.plot(D,B,D,T)
6.  plt.show()
```

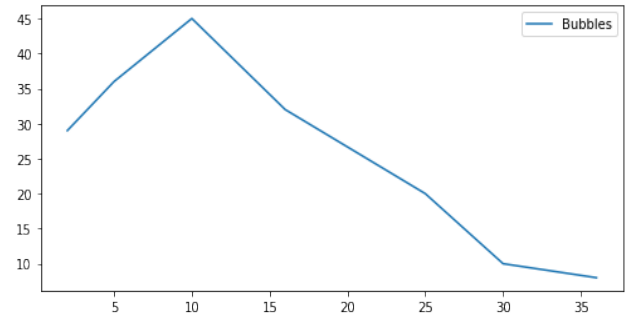


```
1.  D=[2,5,10,16,25,30,36]
2.  B=[29,36,45,32,20,10,8]
3.  T=[30,44,22,10,12,43,2]
4.
5.  plt.plot(D,B)
6.  plt.plot(D,T)
7.  plt.show()
```

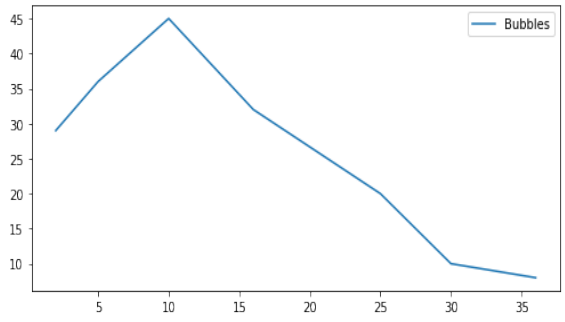


LEGENDS

```
1.  import matplotlib.pyplot as plt
2.  D=[2,5,10,16,25,30,36]
3.  B=[29,36,45,32,20,10,8]
4.  plt.plot(D,B,label="Bubbles")
5.  plt.legend()
6.  plt.show()
```



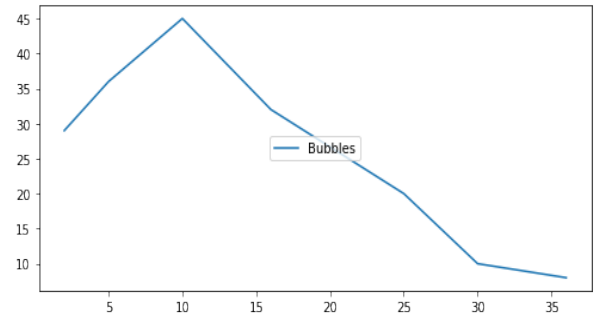

```
1.  import matplotlib.pyplot as plt
2.  D=[2,5,10,16,25,30,36]
3.  B=[29,36,45,32,20,10,8]
4.  plt.plot(D,B)
5.  plt.legend(['Bubbles'])
6.  plt.show()
```



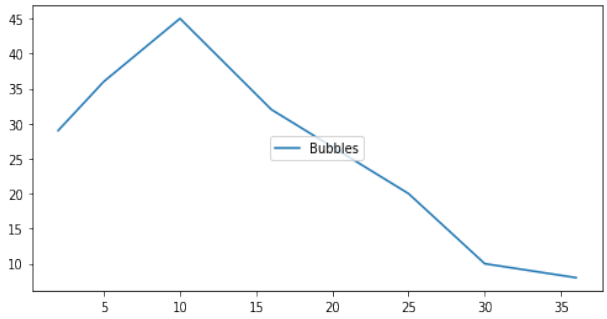
LEGENDS - Locations

Location String	Location Code	Location String	Location Code
'best'	0	'center left'	6
'upper right'	1	'center right'	7
'upper left'	2	'lower center'	8
'lower left'	3	'upper center'	9
'lower right'	4	'center'	10
'right'	5		

```
1.  plt.plot(D,B,label="Bubbles")
2.  plt.legend(loc='center')
```

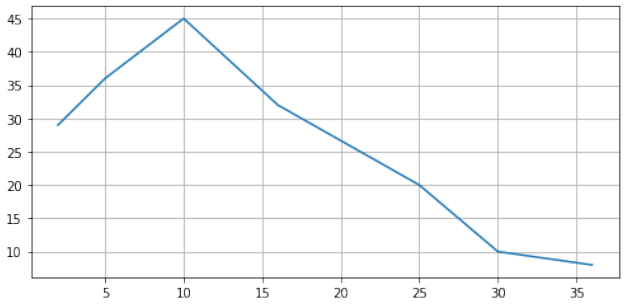



```
1.  plt.plot(D,B,label="Bubbles")
2.  plt.legend(loc=10)
```

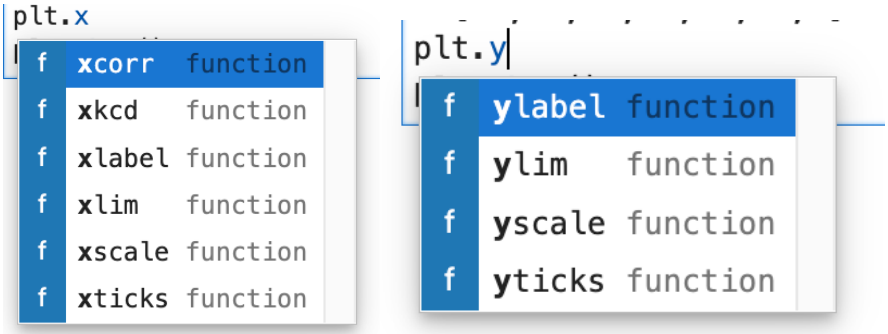


GRID

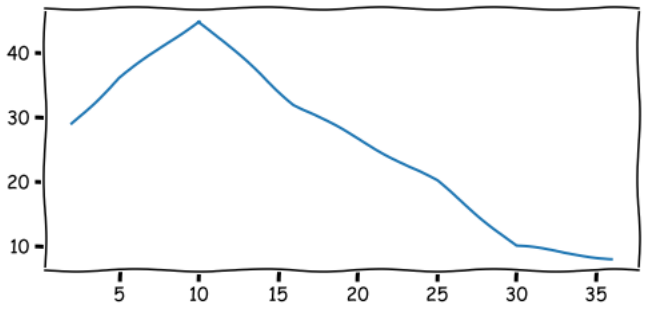
```
1.  plt.grid()
```



x and y axis

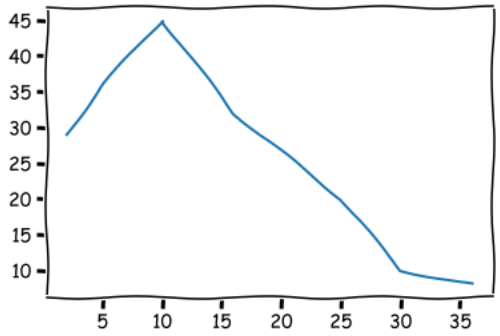


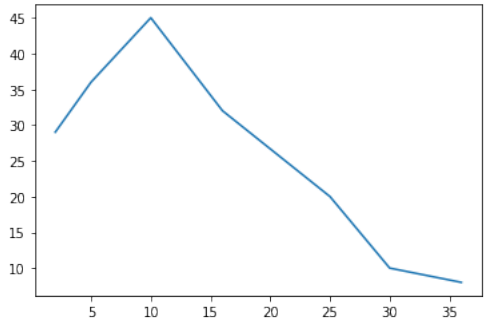
matplotlib.pyplot.xcorr : Plot the cross correlation between *x* and *y*. (info:: https://tinyurl.com/23xe8kfn)
matplotlib.pyplot.xkcd: Turn on xkcd sketch-style drawing mode. (info: https://tinyurl.com/8tnczz3m)



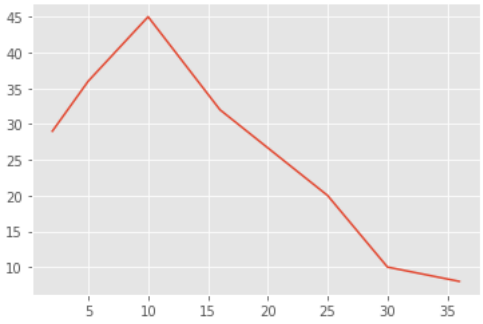
Problem: when we use JupyterLab we will have to reset rcParamsDefault to get the default paramaters. (plt.rcParams.update(plt.rcParamsDefault). This is one way to fix this problem

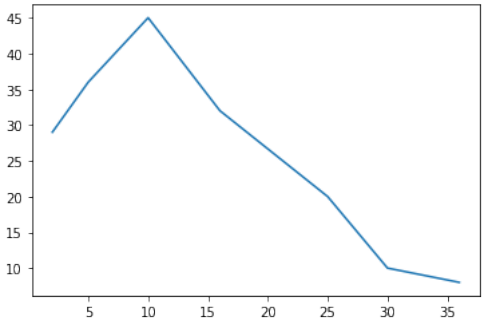
```
1.  D=[2,5,10,16,25,30,36]
2.  B=[29,36,45,32,20,10,8]
3.  with plt.xkcd():
4.      fig1 = plt.figure()
5.      plt.plot(D,B)
6.
7.  fig2 = plt.figure()
8.  plt.plot(D,B)
```





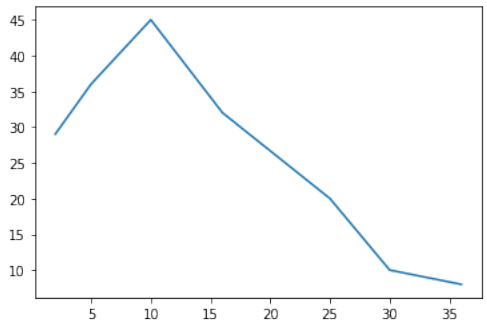

```
1.  D=[2,5,10,16,25,30,36]
2.  B=[29,36,45,32,20,10,8]
3.  with plt.rc_context():
4.      plt.style.use('ggplot')
5.      fig1 = plt.figure()
6.      plt.plot(D,B)
7.
8.  fig2 = plt.figure()
9.  plt.plot(D,B)
```

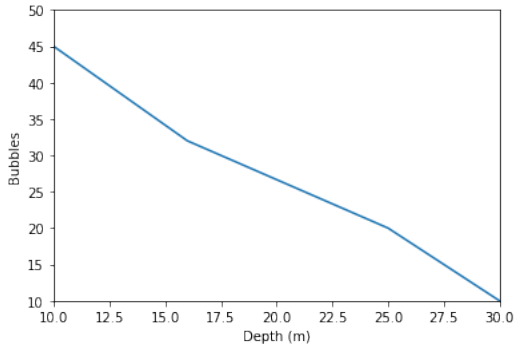




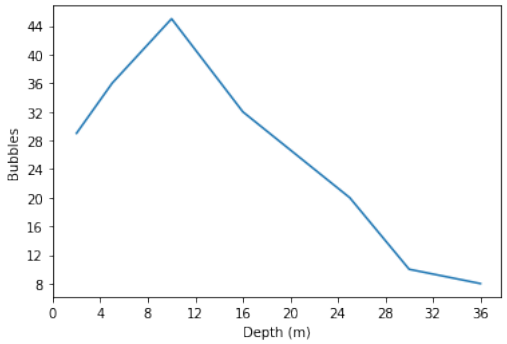
xlim and xticks

```
1. D=[2,5,10,16,25,30,36]
2. B=[29,36,45,32,20,10,8]
3. plt.plot(D,B)
4. plt.xlabel('Depth (m)')
5. plt.ylabel("Bubbles")
6. plt.xlim([10,30])
7. plt.ylim([10,50])
8. plt.show()
```

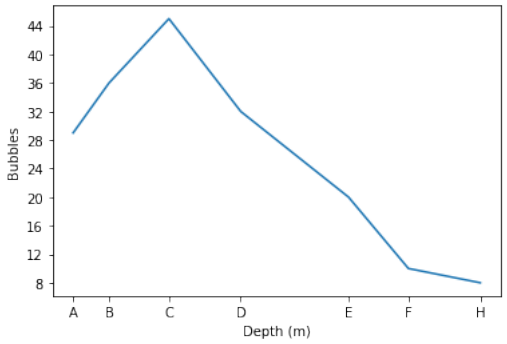




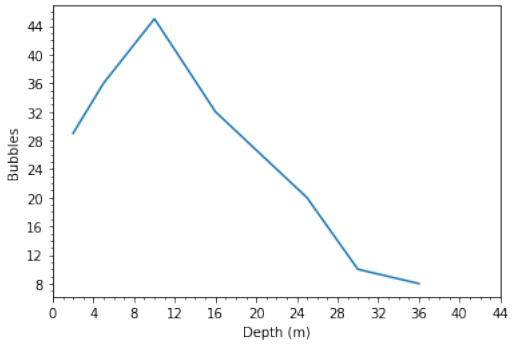

```
1. D=[2,5,10,16,25,30,36]
2. B=[29,36,45,32,20,10,8]
3. plt.plot(D,B)
4. plt.xticks(range(0,40,4))
5. plt.yticks(range(8,46,4))
6. plt.show()
```



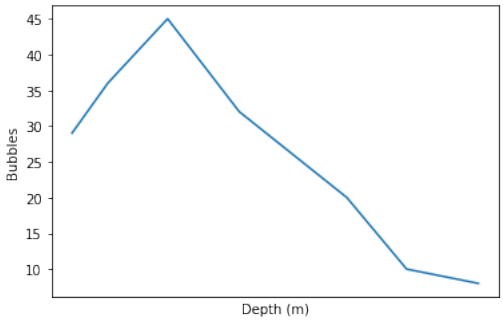

```
1. D=[2,5,10,16,25,30,36]
2. L=['A','B','C','D','E','F','H']
3. plt.xticks(D,L)
```



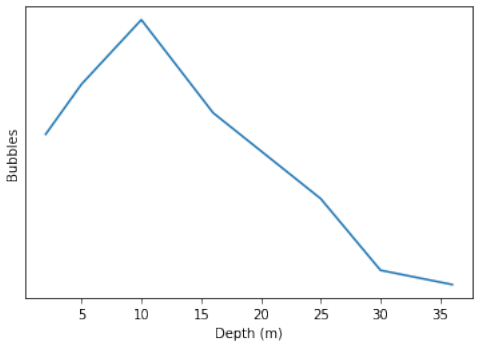

```
1. plt.minorticks_on()
```



```
plt.plot(D,B)
plt.tick_params(
    axis='x',          # changes apply to the x-axis
    which='both',      # both major and minor ticks are affected
    bottom=False,      # ticks along the bottom edge are off
    top=False,         # ticks along the top edge are off
    labelbottom=False) # labels along the bottom edge are off
plt.xlabel('Depth (m)')
plt.ylabel("Bubbles")
```



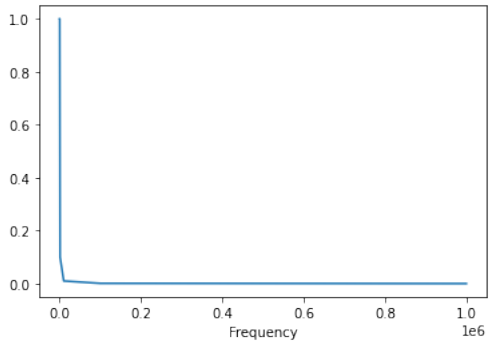

```
plt.plot(D,B)
plt.tick_params(
    axis='y',
    which='both',
    left=False,
    right=False
    labelleft=False)
plt.xlabel('Depth (m)')
plt.ylabel("Bubbles")
plt.show()
```



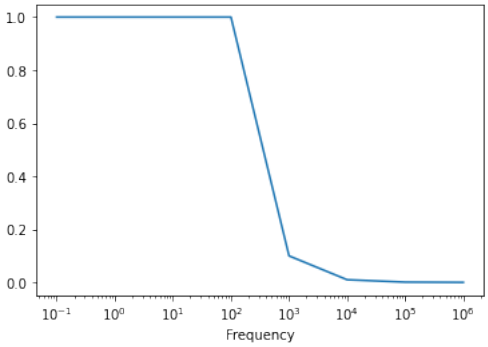

```
plt.tick_params(
    axis='both',          # changes apply to the x-axis
    which='both',      # both major and minor ticks are affected
    bottom=False,      # ticks along the bottom edge are off
    top=False,         # ticks along the top edge are off
    left=False,      # ticks along the bottom edge are off
    right=False,         # ticks along the top edge are off
    labelleft=False, # labels along the bottom edge are off
    labelbottom=False) # labels along the bottom edge are off
```

xscale and yscale

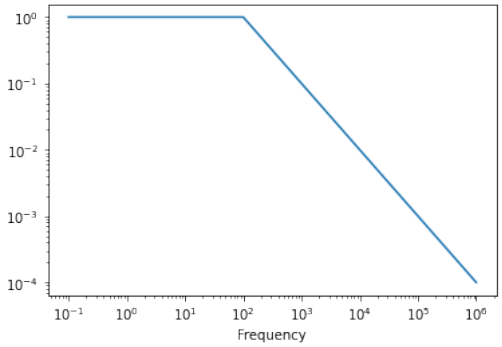
```
1. import matplotlib.pyplot as plt
2. import numpy as np
3. y=np.array([1,1,1,1,1e-1,1e-2,1e-3,1e-4])
4. x=np.array([0.1,1,1e1,1e2,1e3,1e4,1e5,1e6])
5. plt.plot(x, y)
6. plt.xlabel('Frequency')
7. plt.show()
```



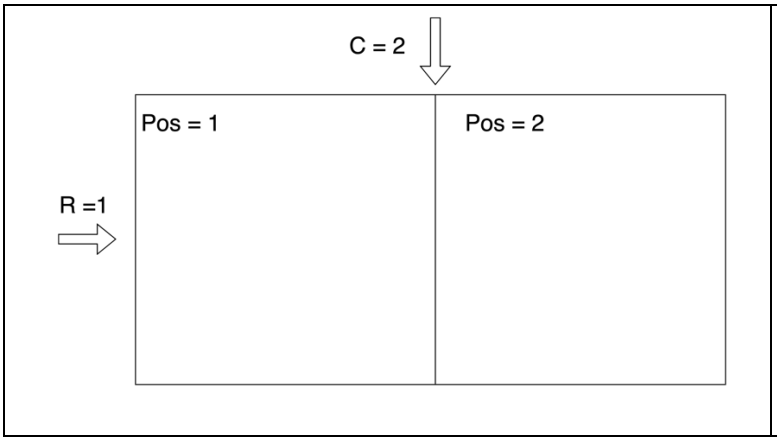

```
1. import matplotlib.pyplot as plt
2. import numpy as np
3. y=np.array([1,1,1,1,1e-1,1e-2,1e-3,1e-4])
4. x=np.array([0.1,1,1e1,1e2,1e3,1e4,1e5,1e6])
5. plt.xscale('log')
6. plt.plot(x, y)
7. plt.xlabel('Frequency')
8. plt.show()
```



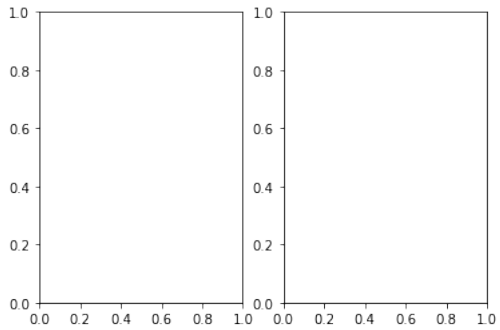

```
1. plt.yscale('log')
```

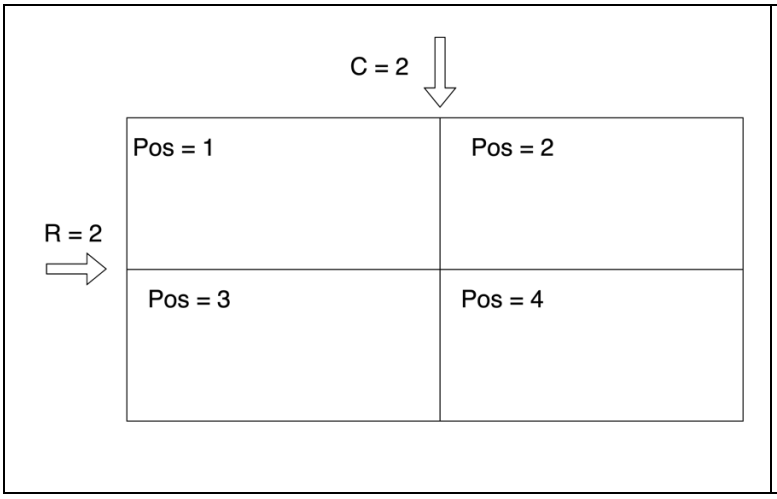


SUBPLOTS
Part I) (R,C,Pos) -> number of subplots (R x C)

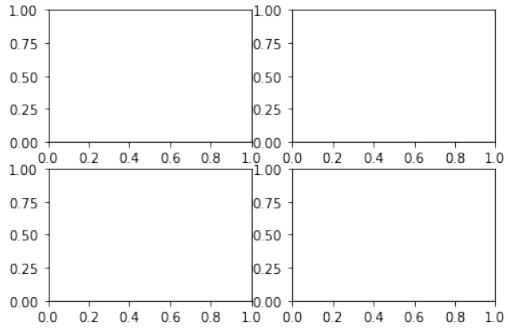



```
1. plt.subplot(1,2,1)
2. plt.subplot(1,2,2)
```

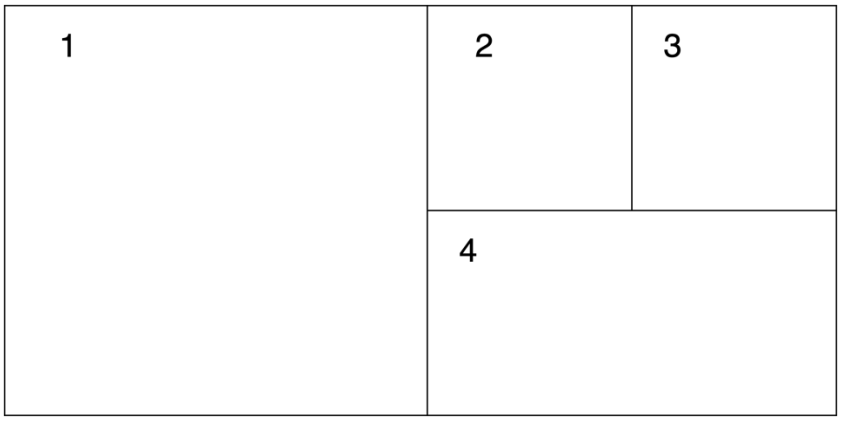




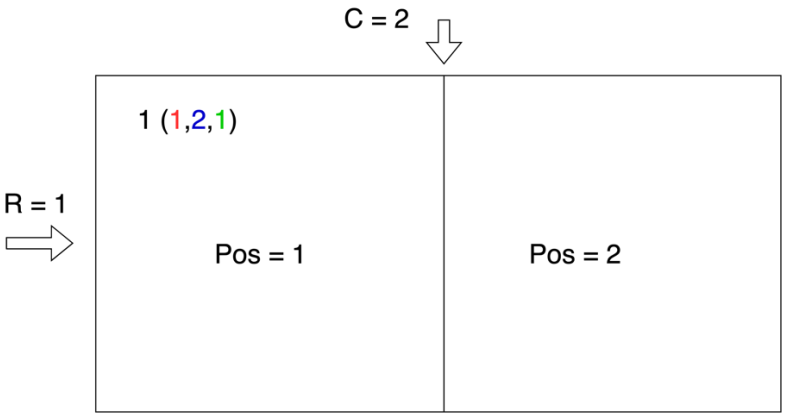

```
1. plt.subplot(2,2,1)
2. plt.subplot(2,2,2)
3. plt.subplot(2,2,3)
4. plt.subplot(2,2,4)
```



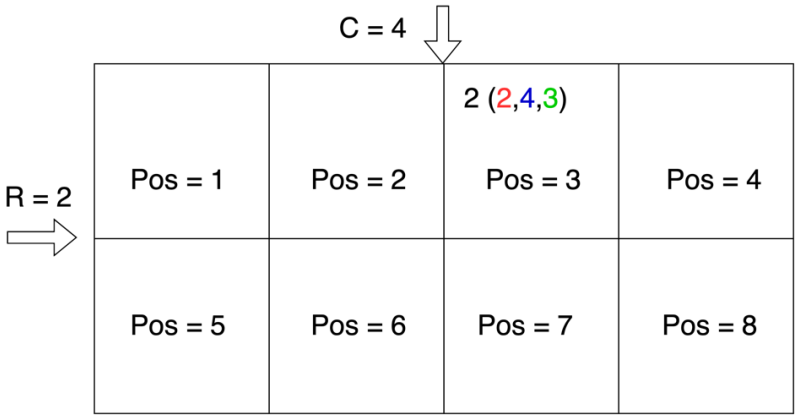
EXAMPLE (MIX DIMENSIONS)



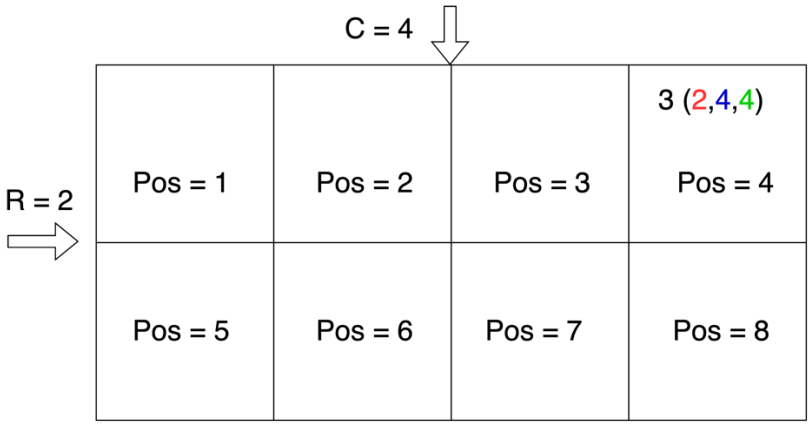
CREATE (1)



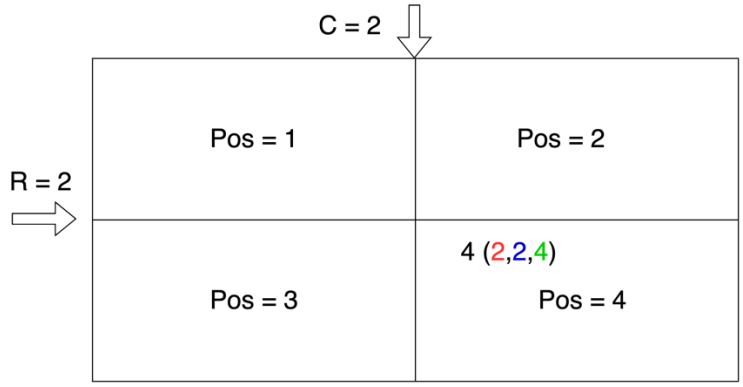
CREATE (2)



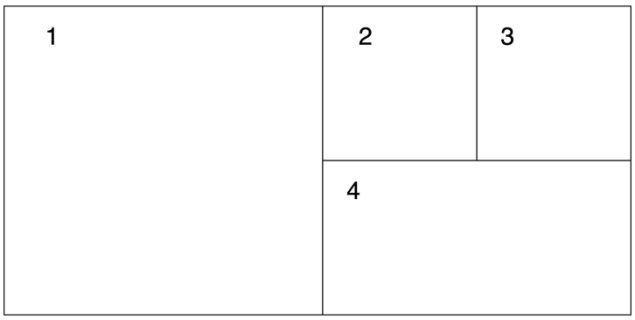
CREATE (3)



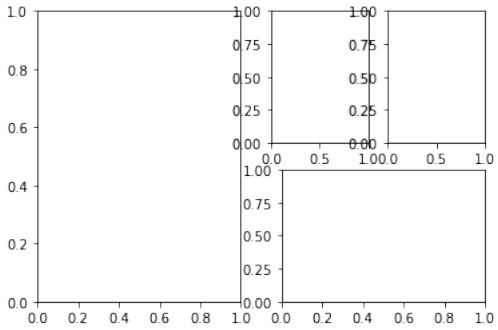
CREATE (4)



RESULT

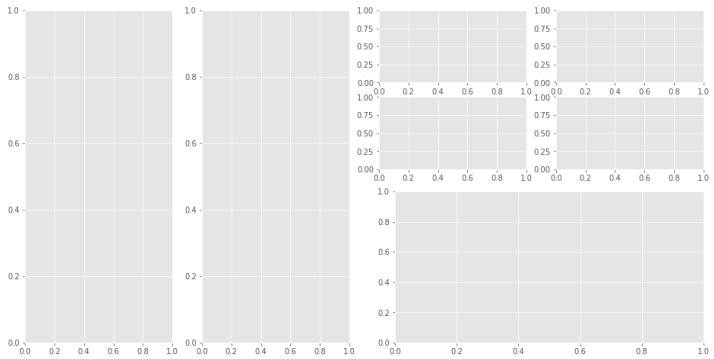



```
1.  plt.subplot(1,2,1)
2.  plt.subplot(2,4,1)
3.  plt.subplot(2,4,4)
4.  plt.subplot(2,2,4)
```

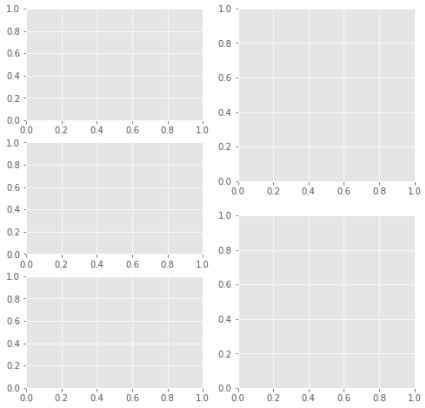


EXAMPLES:

```
1.  fig = plt.figure(figsize=(16,8))
2.  fig.add_subplot(141)   #top left
3.  fig.add_subplot(142)   #top left
4.  fig.add_subplot(443)   #top right
5.  fig.add_subplot(444)   #bottom left
6.  fig.add_subplot(447)   #top right
7.  fig.add_subplot(448)   #bottom left
8.  fig.add_subplot(224)   #bottom right
```

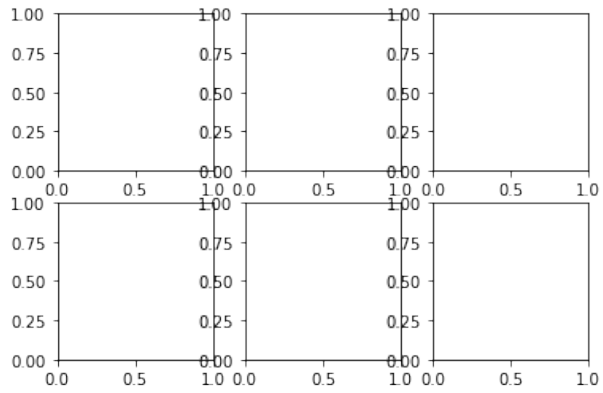


```
1.  plt.figure(figsize=(8,8))
2.  plt.subplot(3,2,1)
3.  plt.subplot(3,2,3)
4.  plt.subplot(3,2,5)
5.  plt.subplot(2,2,2)
6.  plt.subplot(2,2,4)
```

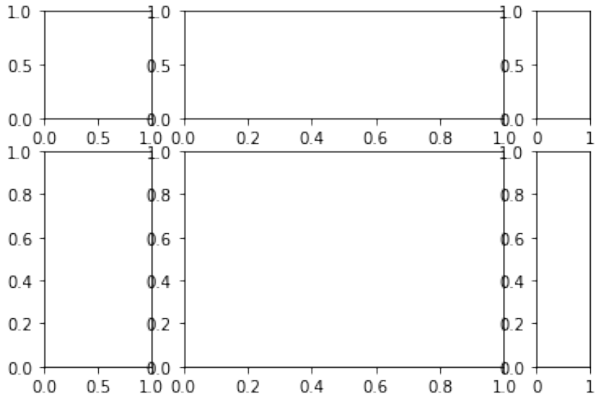


Part II) Using Gridspace

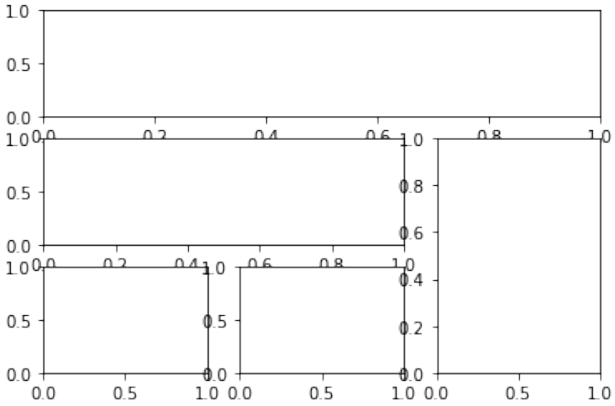
```
1.  import matplotlib
2.  import matplotlib.pyplot as plt
3.  import matplotlib.gridspec as gridspec
4.
5.  gs = gridspec.GridSpec(2, 3)
6.  plt.subplot(gs[0])
7.  plt.subplot(gs[1])
8.  plt.subplot(gs[2])
9.  plt.subplot(gs[3])
10. plt.subplot(gs[4])
11. plt.subplot(gs[5])
```

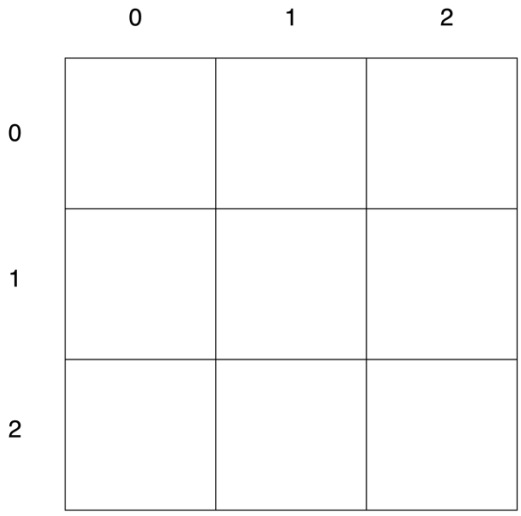


```
1.  import matplotlib
2.  import matplotlib.pyplot as plt
3.  import matplotlib.gridspec as gridspec
4.  gs = gridspec.GridSpec(2, 3,
    width_ratios=[1, 3,.5],height_ratios=[1, 2])
5.  plt.subplot(gs[0])
6.  plt.subplot(gs[1])
7.  plt.subplot(gs[2])
8.  plt.subplot(gs[3])
9.  plt.subplot(gs[4])
10. plt.subplot(gs[5])
```

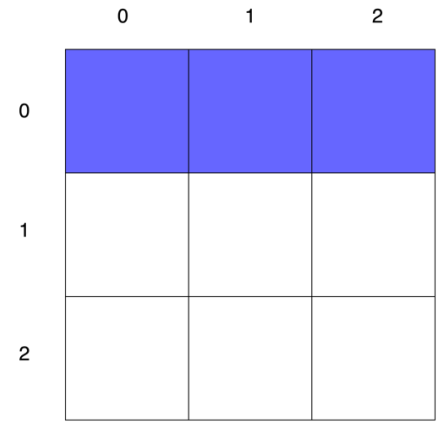


EXAMPLE:

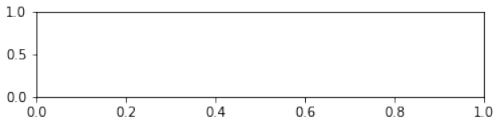




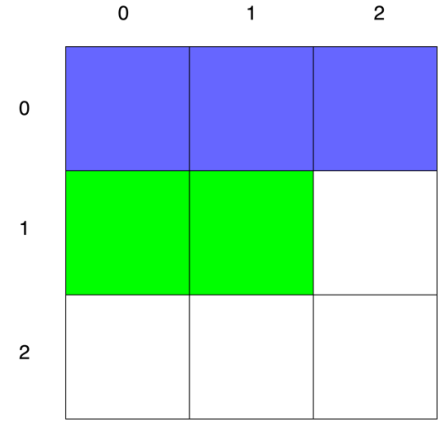
PART I)



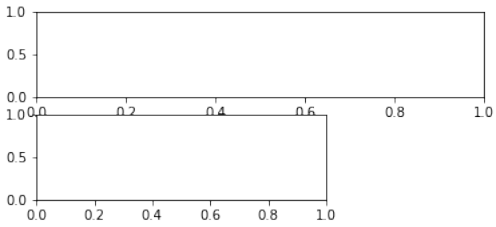

```
1.  import matplotlib
2.  import matplotlib.pyplot as plt
3.  import matplotlib.gridspec as gridspec
4.  gs = gridspec.GridSpec(3, 3)
5.  plt.subplot(gs[0,:])
```



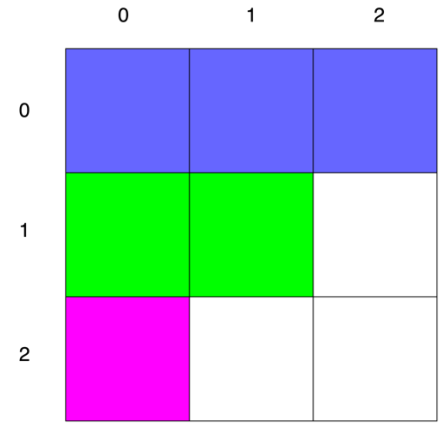
PART II)



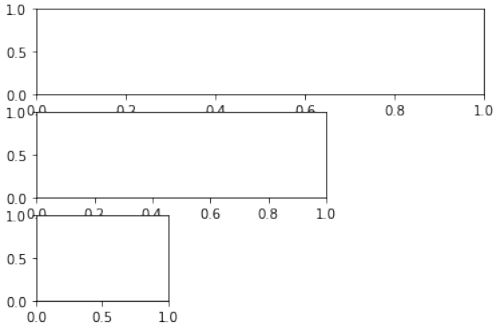

```
1.  import matplotlib
2.  import matplotlib.pyplot as plt
3.  import matplotlib.gridspec as gridspec
4.  gs = gridspec.GridSpec(3, 3)
5.  plt.subplot(gs[0,:])
6.  plt.subplot(gs[0,:])
```



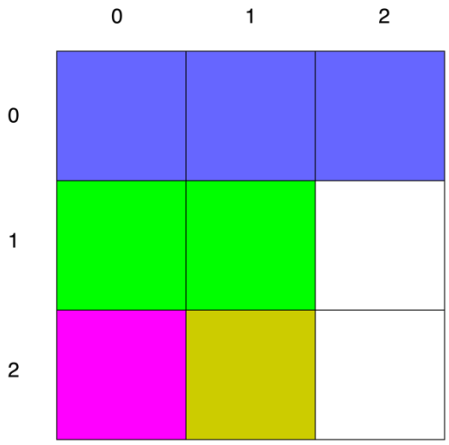
PART III)



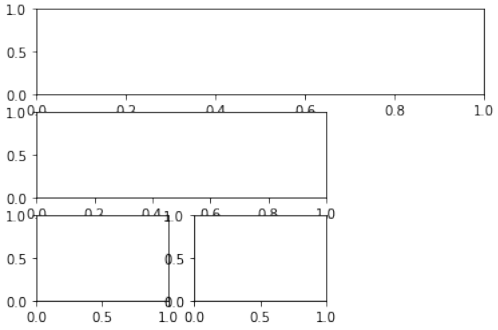

```
1.  import matplotlib
2.  import matplotlib.pyplot as plt
3.  import matplotlib.gridspec as gridspec
4.  gs = gridspec.GridSpec(3, 3)
5.  plt.subplot(gs[0,:])
6.  plt.subplot(gs[0,:])
7.  plt.subplot(gs[2,0])
```



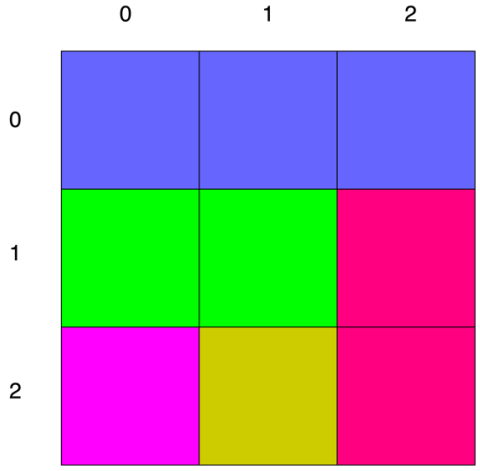
PART IV)



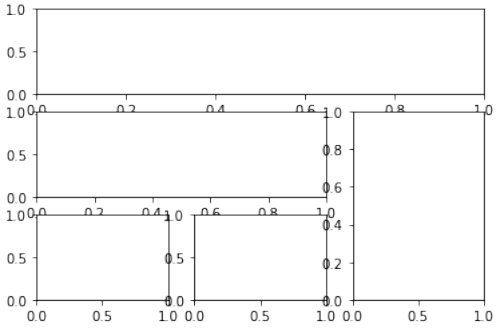

```
1.  import matplotlib
2.  import matplotlib.pyplot as plt
3.  import matplotlib.gridspec as gridspec
4.  gs = gridspec.GridSpec(3, 3)
5.  plt.subplot(gs[0,:])
6.  plt.subplot(gs[0,:])
7.  plt.subplot(gs[2,0])
8.  plt.subplot(gs[2,1])
```



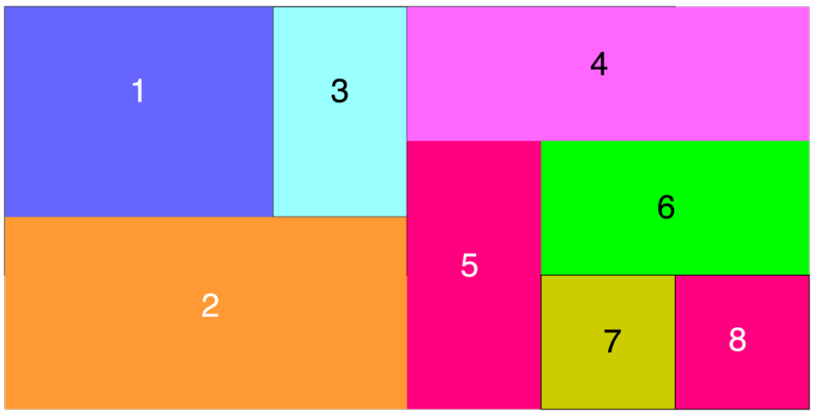
PART V)



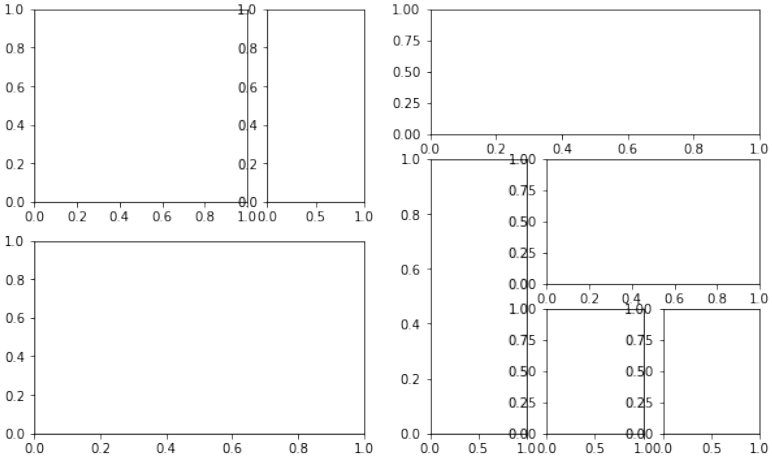

```
1.  import matplotlib
2.  import matplotlib.pyplot as plt
3.  import matplotlib.gridspec as gridspec
4.  gs = gridspec.GridSpec(3, 3)
5.  plt.subplot(gs[0,:])
6.  plt.subplot(gs[0,:])
7.  plt.subplot(gs[2,0])
8.  plt.subplot(gs[2,1])
9.  plt.subplot(gs[1::,2])
```



Example (Advanced)

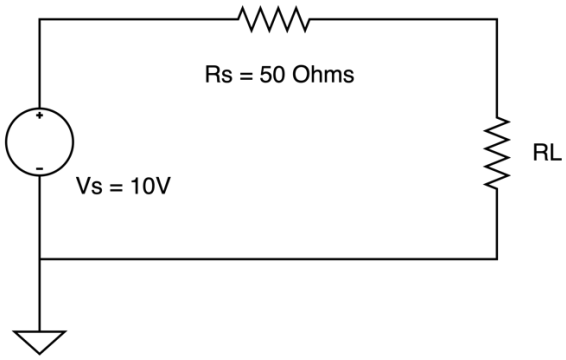


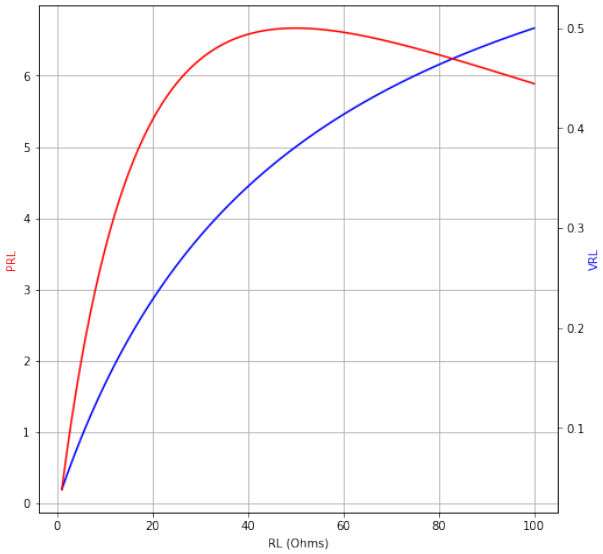

```
1.  import matplotlib
2.  import matplotlib.pyplot as plt
3.  import matplotlib.gridspec as gridspec
4.
5.  plt.figure(figsize=(10,6))
6.  gs = gridspec.GridSpec(1, 2)
7.  gs0=gs[0].subgridspec(2,3)
8.  gs1=gs[1].subgridspec(3,3)
9.
10. plt.subplot(gs0[0,:-1])  #### 1 ####
11. plt.subplot(gs0[1,:])    #### 2 ####
12. plt.subplot(gs0[0,-1])   #### 3 ####
13.
14. plt.subplot(gs1[0,:])    #### 4 ####
15. plt.subplot(gs1[1::,0])  #### 5 ####
16. plt.subplot(gs1[1,1::])  #### 6 ####
17. plt.subplot(gs1[2,1])    #### 7 ####
18. plt.subplot(gs1[2,2])    #### 8 ####
19.
20. plt.show()
```



Example (Power Analysis)

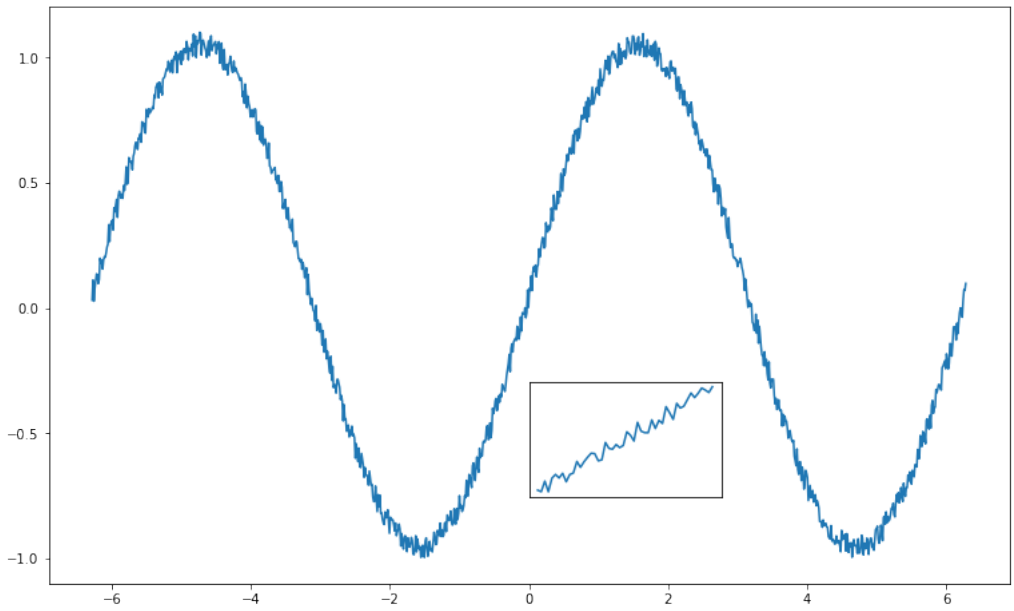
Plot (RL, VRL) and (RL, PRL) for 0 Ohms < RL < 101 Ohms using the circuit below.






```
#Vs =10V, Rs=50 Ohms, RL=1-100Ohms
#VRL=(Vs)RL/(RS+RL)
#IRL=VRL/RL
# PRL = IRL*VRL
Vs=10;
Rs=50;

VRL=[]
IRL=[]
PRL = []

for i in range(1,101):
    Ic=Vs/(i+Rs)
    Vc = Ic*i
    Pc=Ic*Vc
    VRL.append(Vc)
    IRL.append(Ic)
    PRL.append(Pc)

fig=plt.figure(figsize=(8,8))
ax1 = fig.add_subplot(1, 1, 1)
ax2 = ax1.twinx()
ax1.plot(RL, VRL, 'b-')
ax2.plot(RL, PRL, 'r-')
ax1.set_ylabel('PRL', color='red')
ax2.set_ylabel('VRL', color='blue')
ax1.set_xlabel('RL (Ohms)')
ax1.grid()
```

Part III) Using add_axes




```
1.   fig=plt.figure(figsize=(10,6))
2.  ax1=fig.add_axes([0.0,0.0,1,1])
3.  x=np.linspace(-2*np.pi,2*np.pi,1000)
4.  noise=np.random.random(len(x))
5.  y=np.sin(x)+noise/10
6.  plt.plot(x,y)
7.
8.  ax2=fig.add_axes([0.5,0.15,0.2,0.2])
9.  plt.plot(x[500:550],y[500:550])
10. plt.tick_params(
11.     axis='both',          # changes apply to the x-axis
12.     which='both',      # both major and minor ticks are affected
13.     bottom=False,      # ticks along the bottom edge are off
14.     top=False,         # ticks along the top edge are off
15.     left=False,      # ticks along the bottom edge are off
16.     right=False,         # ticks along the top edge are off
17.     labelleft=False, # labels along the bottom edge are off
18.     labelbottom=False) # labels along the bottom edge are off
```