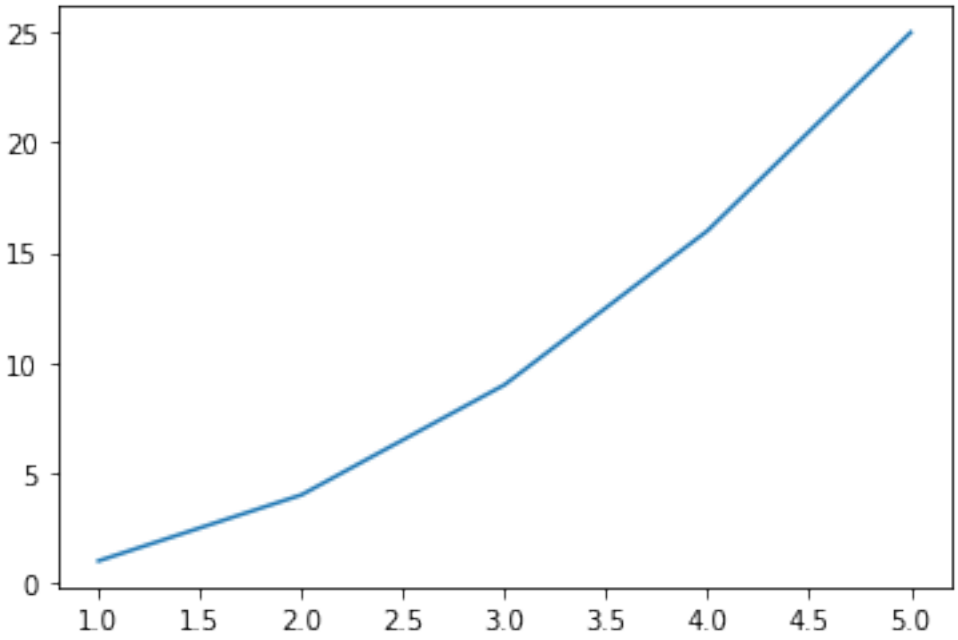
Untitled

March 10, 2021

0.0.1 Some simple examples

0.0.2 line plot

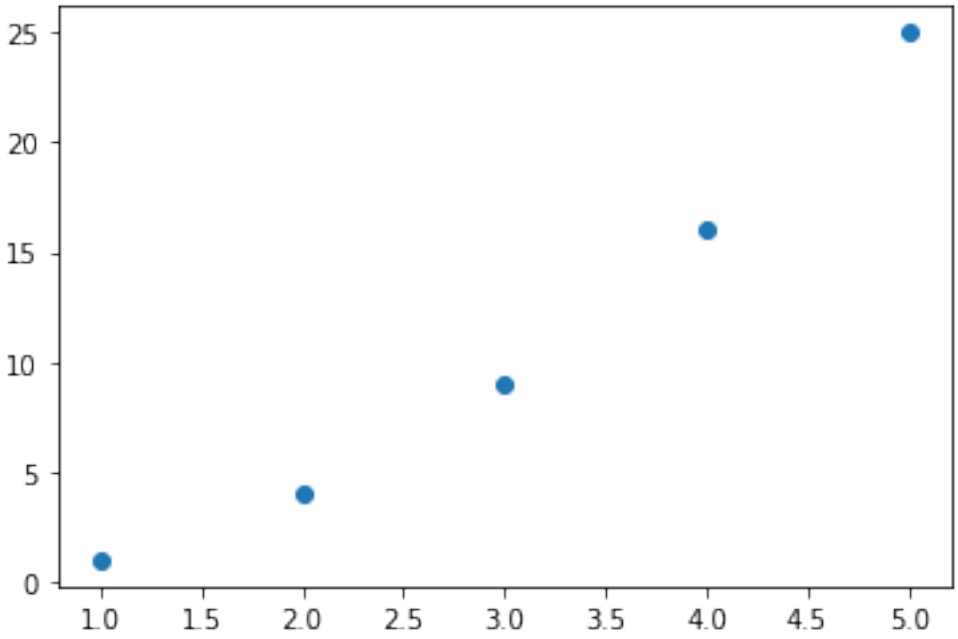
[1]:
```
import matplotlib.pyplot as plt
x = [1,2,3,4,5]
y = [1,4,9,16,25]
plt.plot(x,y)
plt.show()
```



0.0.3 scatter plot

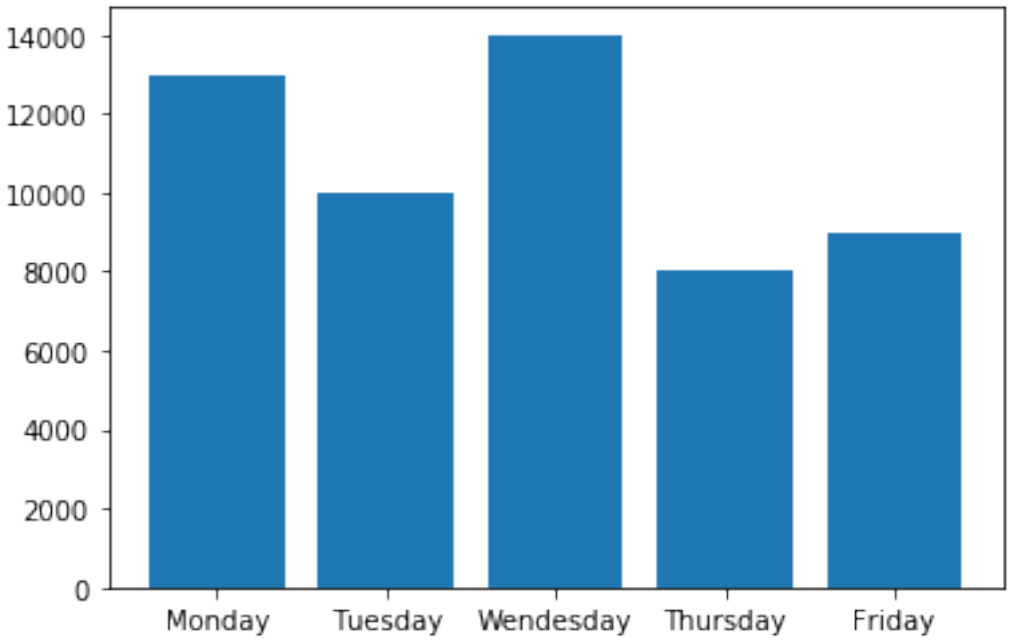
[2]:
```
import matplotlib.pyplot as plt
x = [1,2,3,4,5]
y = [1,4,9,16,25]
```

```
plt.scatter(x,y)
plt.show()
```



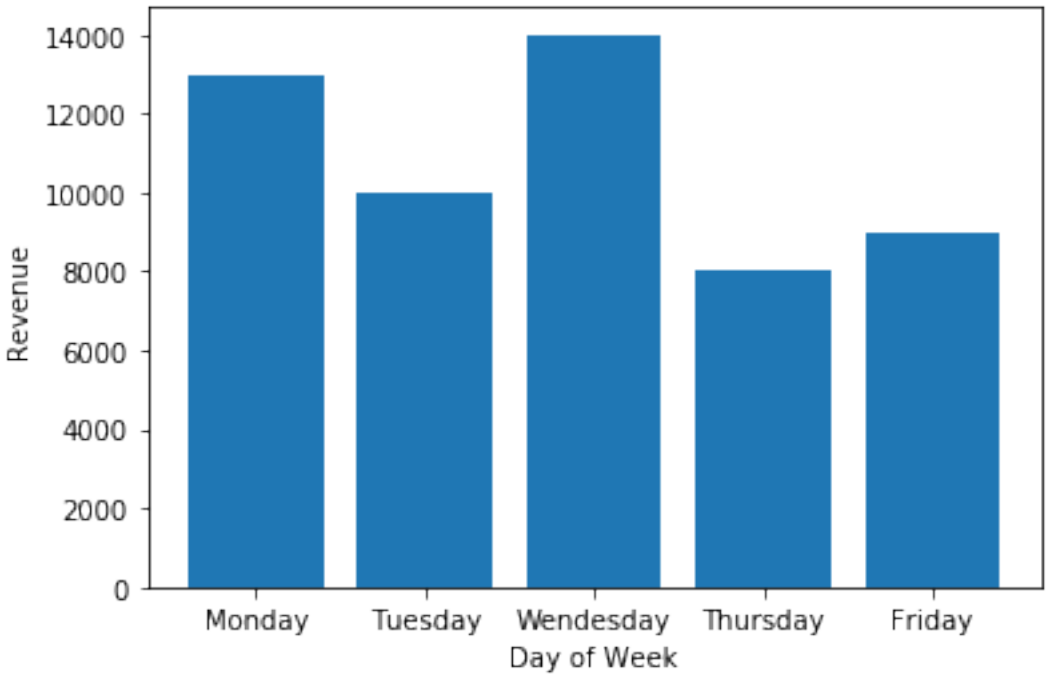
0.0.4 bar plot

[3]:
```
days = ['Monday','Tuesday','Wendesday','Thursday','Friday']
revenue = [13000,10000,14000,8000,9000]
plt.bar(days,revenue)
plt.show()
```



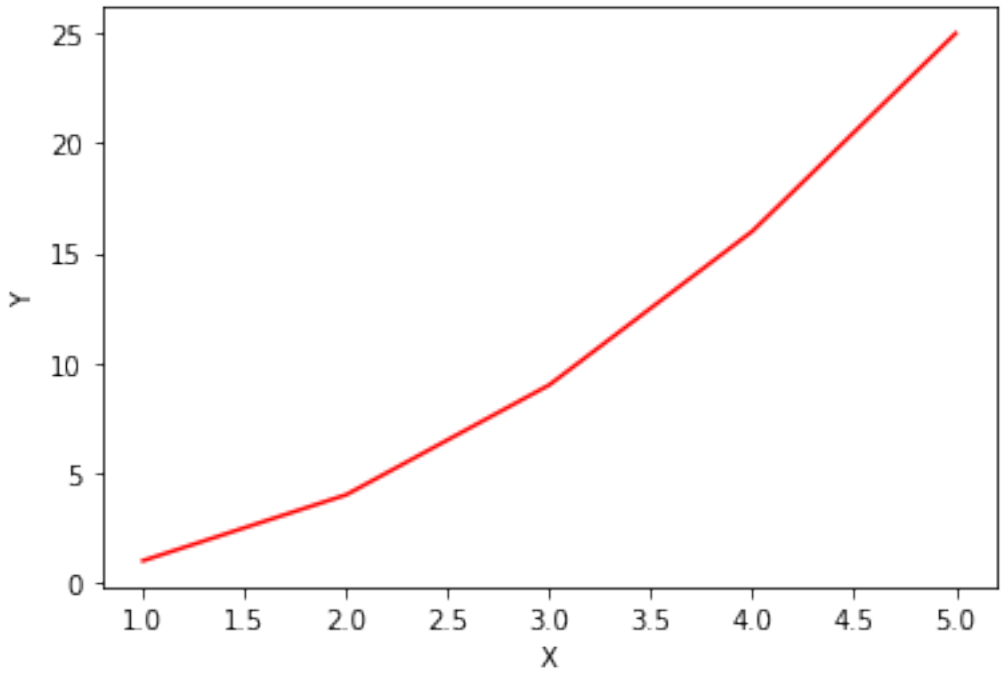
0.0.5 add axis labels

[4]:
```
days = ['Monday','Tuesday','Wendesday','Thursday','Friday']
revenue = [13000,10000,14000,8000,9000]
plt.bar(days,revenue)
plt.xlabel('Day of Week')
plt.ylabel('Revenue')
plt.show()
```

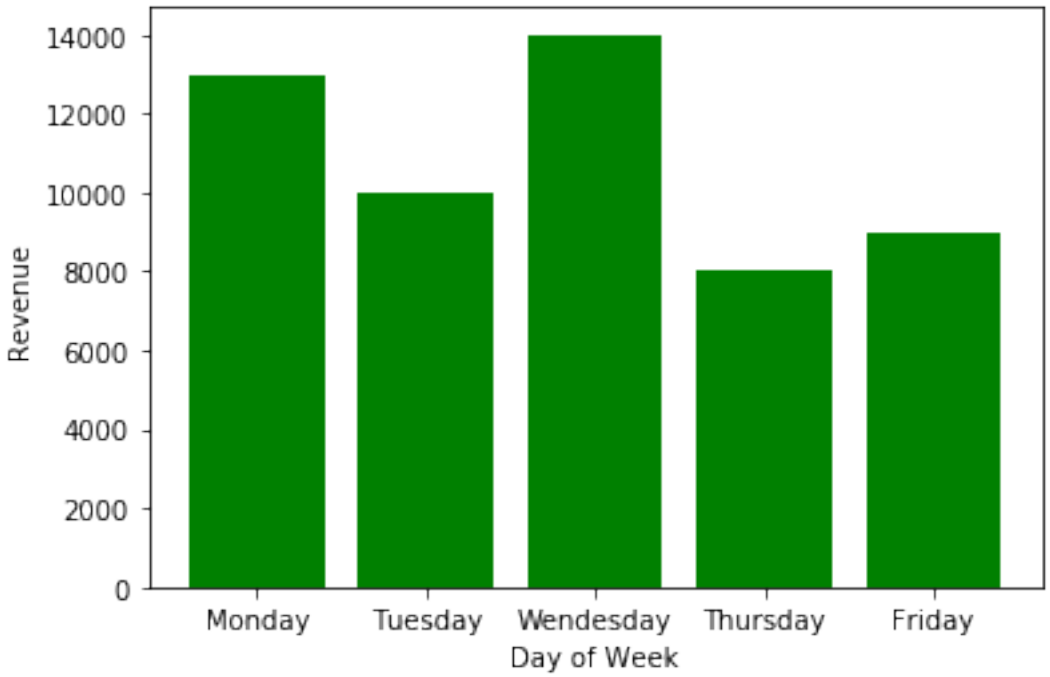


0.0.6 Dont like the color? use the color= option

[7]:
```
import matplotlib.pyplot as plt
x = [1,2,3,4,5]
y = [1,4,9,16,25]
plt.plot(x,y,color='red')
plt.xlabel('X')
plt.ylabel('Y')
plt.show()
```



[8]:
```
days = ['Monday','Tuesday','Wendesday','Thursday','Friday']
revenue = [13000,10000,14000,8000,9000]
plt.bar(days,revenue,color='green')
plt.xlabel('Day of Week')
plt.ylabel('Revenue')
plt.show()
```



0.0.7 That's the gist of it

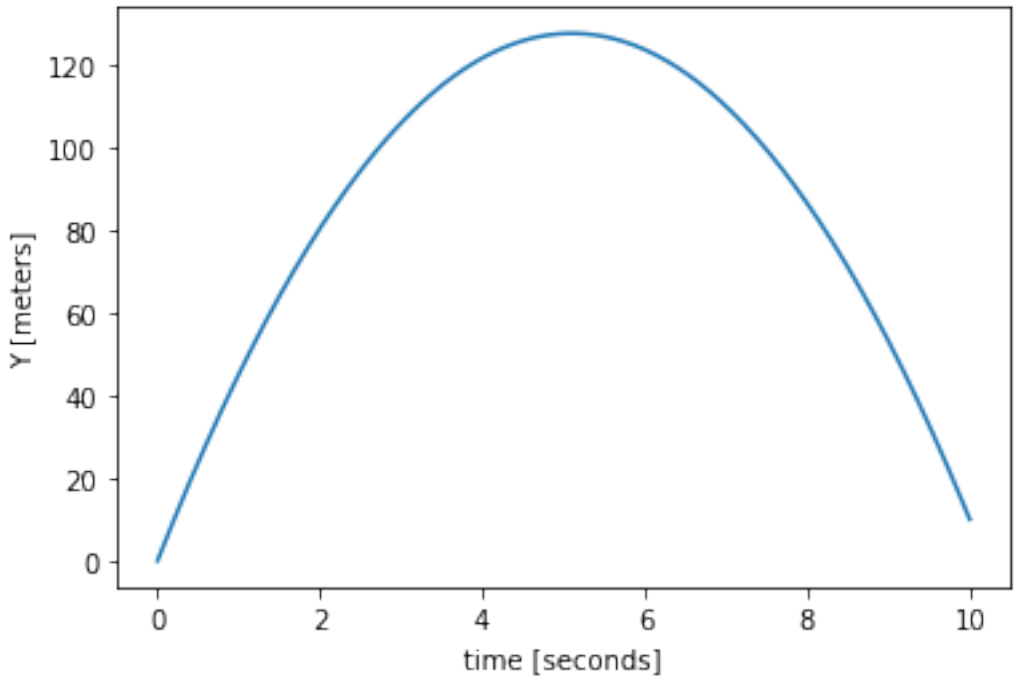
See here: https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html#matplotlib.pyplot.plot

0.0.8 Example: Plotting projectile motion

Let's write a program to plot the motion of a projectile

[31]:
```
import math

g = 9.8

def y(t,vi,theta,yi=0):
    result = yi + vi * math.sin(theta) * t - 0.5 * g * t**2
    return result
def x(t,vi,theta,xi=0):
    result = xi + vi * math.cos(theta) * t
    return result
def vy(t,vi,theta):
    result = vi - g * t
    return result

theta = math.radians(30)
vi = 100
```

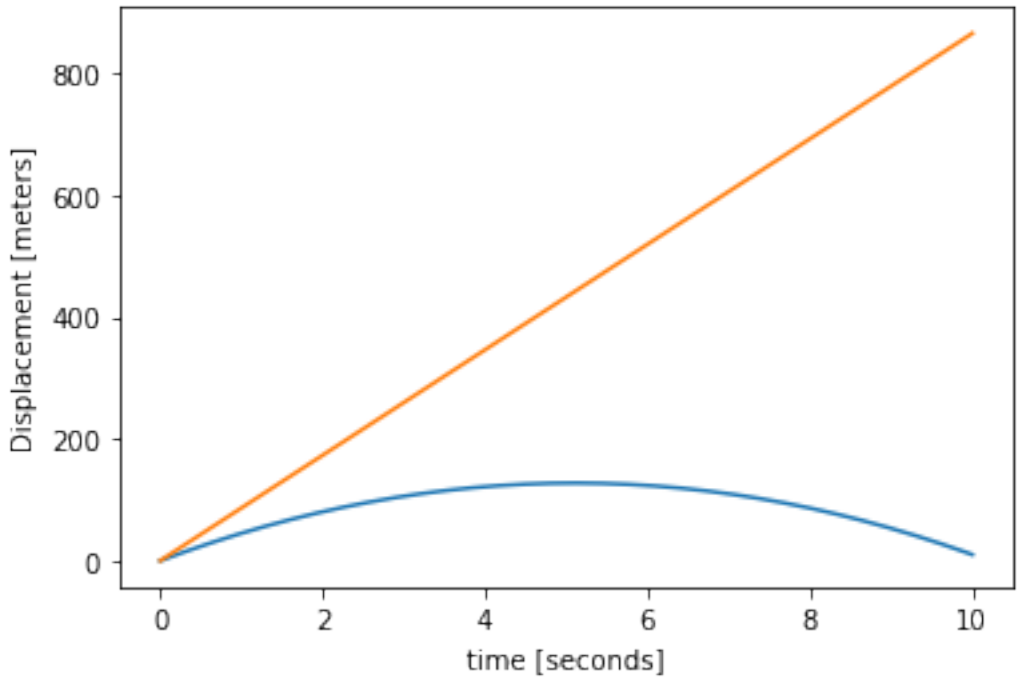
```
time = []
yvals = []
xvals = []
vyvals = []
t = 0
while t < 10:
    time.append(t)
    yvals.append(y(t,vi,theta))
    xvals.append(x(t,vi,theta))
    vyvals.append(vy(t,vi,theta))
    t+=0.01

plt.plot(time,yvals)
plt.xlabel('time [seconds]')
plt.ylabel('Y [meters]')
plt.show()
```



0.0.9 Plotting multiple series at once

[32]:
```
import math

g = 9.8
```

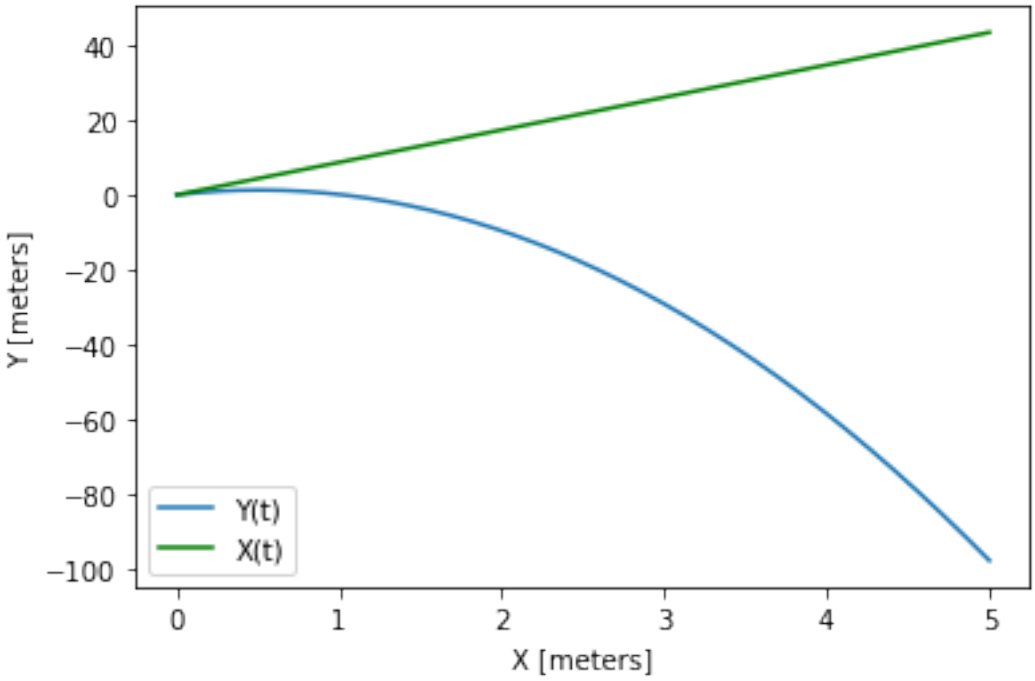
```
def y(t,vi,theta,yi=0):
    result = yi + vi * math.sin(theta) * t - 0.5 * g * t**2
    return result
def x(t,vi,theta,xi=0):
    result = xi + vi * math.cos(theta) * t
    return result
def vy(t,vi,theta):
    result = vi - g * t
    return result

theta = math.radians(30)
vi = 100
time = []
yvals = []
xvals = []
vyvals = []
t = 0
while t < 10:
    time.append(t)
    yvals.append(y(t,vi,theta))
    xvals.append(x(t,vi,theta))
    vyvals.append(vy(t,vi,theta))
    t+=0.01

plt.plot(time,yvals)
plt.plot(time,xvals)
plt.xlabel('time [seconds]')
plt.ylabel('Displacement [meters]')
plt.show()
```



0.0.10 Add a legend

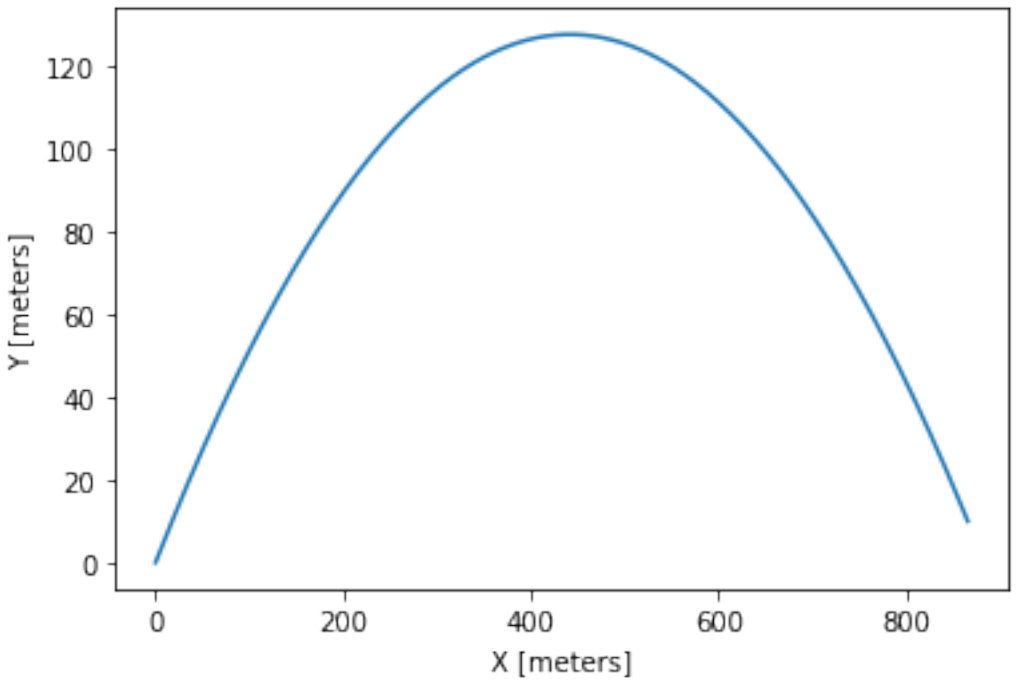
[41]:
```
import math

g = 9.8

def y(t,vi,theta,yi=0):
    result = yi + vi * math.sin(theta) * t - 0.5 * g * t**2
    return result
def x(t,vi,theta,xi=0):
    result = xi + vi * math.cos(theta) * t
    return result
def vy(t,vi,theta):
    result = vi - g * t
    return result

theta = math.radians(30)
vi = 10
time = []
yvals = []
xvals = []
vyvals = []
t = 0
```

```
while t < 5:
    time.append(t)
    yvals.append(y(t,vi,theta))
    xvals.append(x(t,vi,theta))
    vyvals.append(vy(t,vi,theta))
    t+=0.01

plt.plot(time,yvals,label='Y(t)')
plt.xlabel('X [meters]')
plt.ylabel('Y [meters]')
plt.plot(time,xvals,color='green',label='X(t)')
plt.legend(loc='lower left')
plt.show()
```

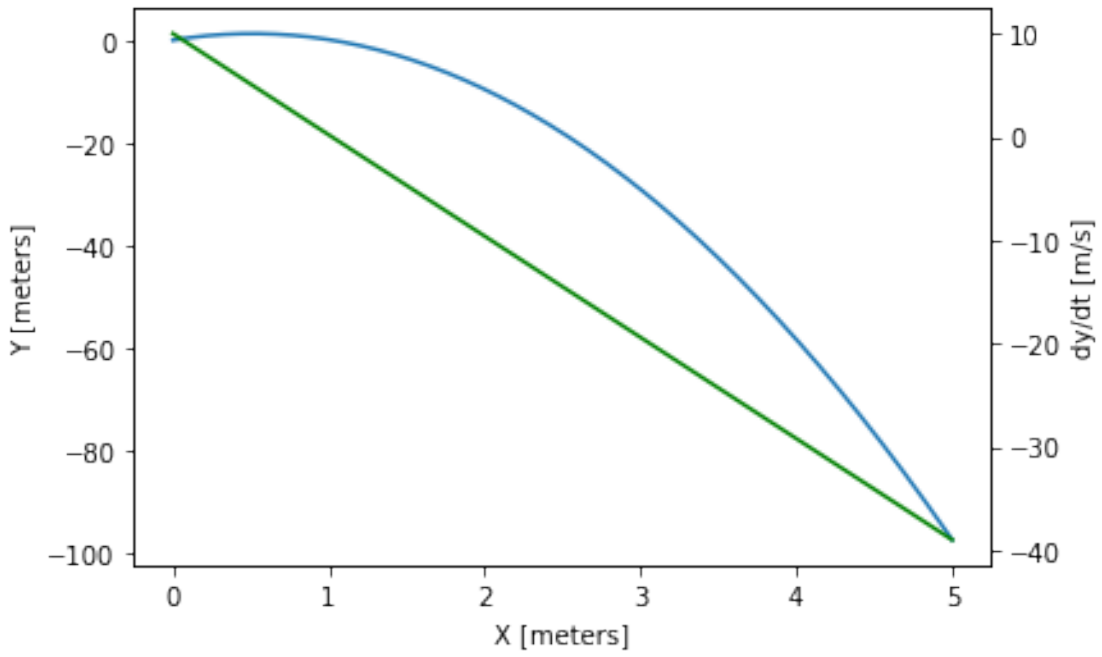


[33]:
```
import math

g = 9.8

def y(t,vi,theta,yi=0):
    result = yi + vi * math.sin(theta) * t - 0.5 * g * t**2
    return result
def x(t,vi,theta,xi=0):
    result = xi + vi * math.cos(theta) * t
    return result
```

```
def vy(t,vi,theta):
    result = vi - g * t
    return result

theta = math.radians(30)
vi = 100
time = []
yvals = []
xvals = []
vyvals = []
t = 0
while t < 10:
    time.append(t)
    yvals.append(y(t,vi,theta))
    xvals.append(x(t,vi,theta))
    vyvals.append(vy(t,vi,theta))
    t+=0.01

plt.plot(xvals,yvals)
plt.xlabel('X [meters]')
plt.ylabel('Y [meters]')
plt.show()
```



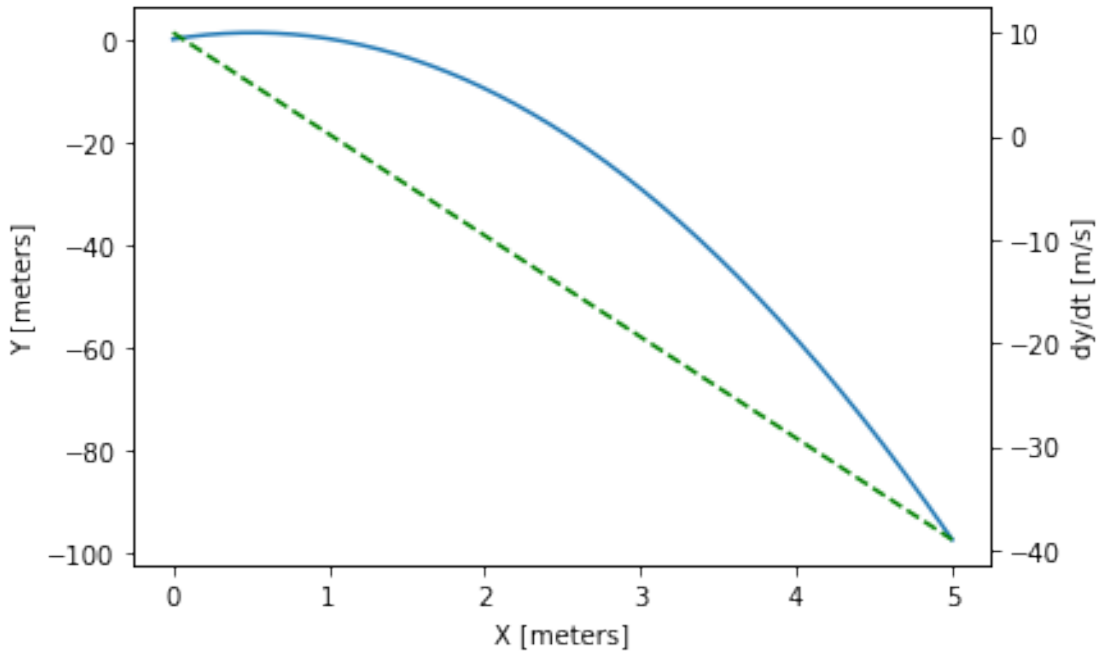
[34]:
```
import math

g = 9.8

def y(t,vi,theta,yi=0):
    result = yi + vi * math.sin(theta) * t - 0.5 * g * t**2
    return result
def x(t,vi,theta,xi=0):
    result = xi + vi * math.cos(theta) * t
    return result
def vy(t,vi,theta):
    result = vi - g * t
    return result

theta = math.radians(30)
vi = 10
time = []
yvals = []
xvals = []
vyvals = []
t = 0
while t < 5:
    time.append(t)
    yvals.append(y(t,vi,theta))
    xvals.append(x(t,vi,theta))
    vyvals.append(vy(t,vi,theta))
    t+=0.01

plt.plot(time,yvals)
plt.xlabel('X [meters]')
plt.ylabel('Y [meters]')
plt.twinx()
plt.plot(time,vyvals,color='green')
plt.ylabel('dy/dt [m/s]')
plt.show()
```



0.0.11 Change the linestyle

[38]:
```
import math

g = 9.8

def y(t,vi,theta,yi=0):
    result = yi + vi * math.sin(theta) * t - 0.5 * g * t**2
    return result
def x(t,vi,theta,xi=0):
    result = xi + vi * math.cos(theta) * t
    return result
def vy(t,vi,theta):
    result = vi - g * t
    return result

theta = math.radians(30)
vi = 10
time = []
yvals = []
xvals = []
vyvals = []
t = 0
while t < 5:
```

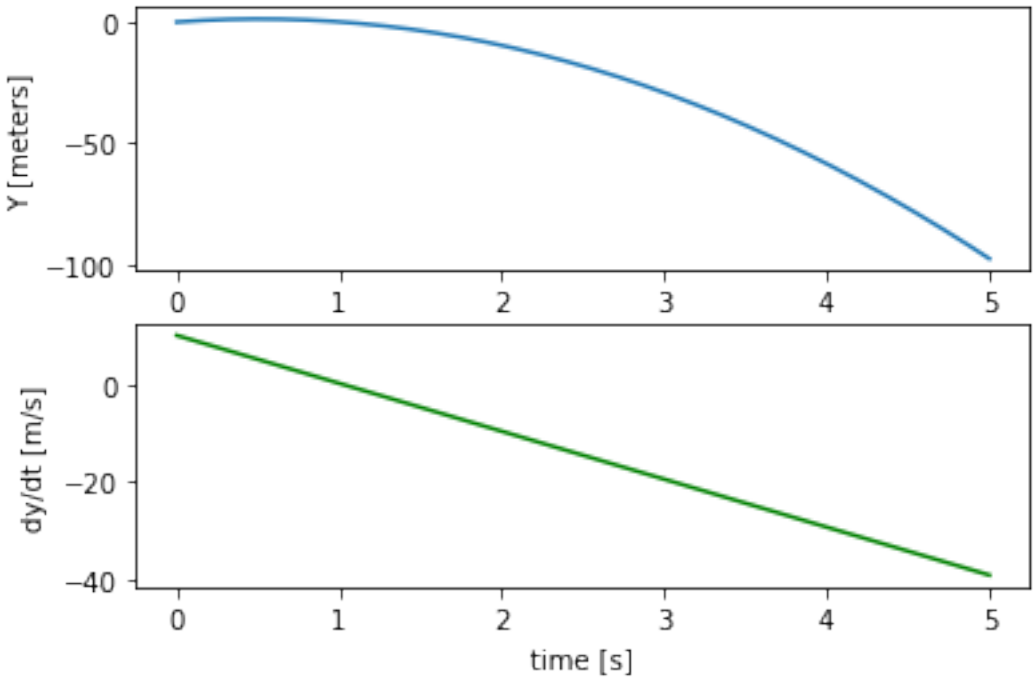
```
    time.append(t)
    yvals.append(y(t,vi,theta))
    xvals.append(x(t,vi,theta))
    vyvals.append(vy(t,vi,theta))
    t+=0.01

plt.plot(time,yvals)
plt.xlabel('X [meters]')
plt.ylabel('Y [meters]')
plt.twinx()
plt.plot(time,vyvals,color='green',linestyle='--')
plt.ylabel('dy/dt [m/s]')
plt.show()
```



0.0.12 subplots

[42]:
```
import math

g = 9.8

def y(t,vi,theta,yi=0):
    result = yi + vi * math.sin(theta) * t - 0.5 * g * t**2
    return result
def x(t,vi,theta,xi=0):
```

```
    result = xi + vi * math.cos(theta) * t
    return result
def vy(t,vi,theta):
    result = vi - g * t
    return result

theta = math.radians(30)
vi = 10
time = []
yvals = []
xvals = []
vyvals = []
t = 0
while t < 5:
    time.append(t)
    yvals.append(y(t,vi,theta))
    xvals.append(x(t,vi,theta))
    vyvals.append(vy(t,vi,theta))
    t+=0.01

plt.subplot(2,1,1) #numrows, numcols, index
plt.plot(time,yvals)
plt.xlabel('X [meters]')
plt.ylabel('Y [meters]')
plt.subplot(2,1,2)
plt.plot(time,vyvals,color='green')
plt.xlabel('time [s]')
plt.ylabel('dy/dt [m/s]')
plt.show()
```



0.0.13 saving with plt.savefig()

[43]:
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon Mar  8 06:25:26 2021

@author: tjwilli
"""

"""
Author: T Williamson
Date:   March 8 2021
projectile2.py: a program to find the angle for max horizontal
    displacement given an initial velocity and plot the results
"""

import math
import matplotlib.pyplot as plt

#grav accel on surface of Earth
G = 9.81 #m/s2
```

```
def time_of_flight(theta,vi,yi):
    """
    Given launch angle "theta" [radians], initial velocity "vi"
    [m/s], and  initial height "yi" [m], calculate
    the time of flight for a launched projectile
    """
    vy = vi * math.sin(theta)
    tof = ( vy + math.sqrt(vy**2 + 2*G*yi) ) / G
    return tof


def xrange(theta,vi,yi):
    """
    Given launch angle "theta" [radians], initial velocity "vi"
    [m/s], and  initial height "yi" [m], calculate
    the final horizontal range
    """
    vx = vi * math.cos(theta)
    tof = time_of_flight(theta,vi,yi)
    return vx * tof


def find_max(vi,yi,theta_min=0,theta_max=90,dtheta=0.1):
    """
    Given initial velocity and height, find theta which maximizes xrange
    (over the range [theta_min,theta_max] )
    """
    #Eventual max xrange, and corresponding theta
    #Start at invalid value so the first loop replaces them
    xmax = -9999
    theta_of_xmax = -9999

    theta_vals = []
    x_vals = []
    theta = theta_min
    while theta <= theta_max:
        delta_x = xrange(math.radians(theta),vi,yi)
        #Keep collection of theta and x for later plotting
        theta_vals.append(theta)
        x_vals.append(delta_x)

        if delta_x > xmax:
            xmax = delta_x
            theta_of_xmax = theta
        theta += dtheta
    return theta_of_xmax, xmax, theta_vals, x_vals
```

```
def main():
    vi = float(input("What is the initial velocity [m/s]? "))
    yi = float(input("What is the initial height [m]? "))
    theta,xmax,tvals,xvals = find_max(vi,yi)
    print( "Horizontal displacement is maximized at an angle␣
 ↪of",theta,"degrees",\
                "with a value of",xmax,"meters.")

    plt.plot(tvals,xvals)
    plt.xlabel("angle [deg]")
    plt.ylabel("range [m]")

    #Plot a dashed line at the maximum
    label = 'x_max = {:.1f} m \n@ theta = {:.1f} deg'.format(xmax,theta)
    plt.axvline(theta,ls='--',c='gray',label=label)
    plt.legend()
    plt.title('xrange vs theta (vi={},yi={})'.format(vi,yi))
    plt.show()

#main()
```

[ ]: