Mathematical Economics

Alpha Chiang

Chapter 3

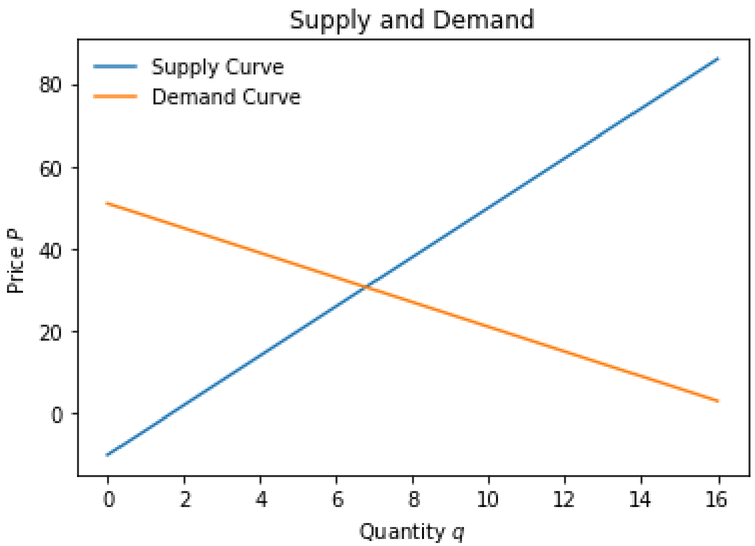
3.2 Partial Market Equilibrium—A Linear Model

In [23]:
```
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import sympy as sy
```

In [24]:
```
def S(P,c=10,d=6):
    return (-c + d*P)

def D(P,a=51,b=3):
    return (a - b*P)

P = np.linspace(0, 16, 1000)
```

In [25]:
```
plt.plot(P, S(P), label = "Supply Curve")
plt.plot(P, D(P), label = "Demand Curve")
plt.title("Supply and Demand")
plt.legend(frameon = False)
plt.xlabel("Quantity $q$")
plt.ylabel("Price $P$")
```

Out[25]:
```
Text(0, 0.5, 'Price $P$')
```



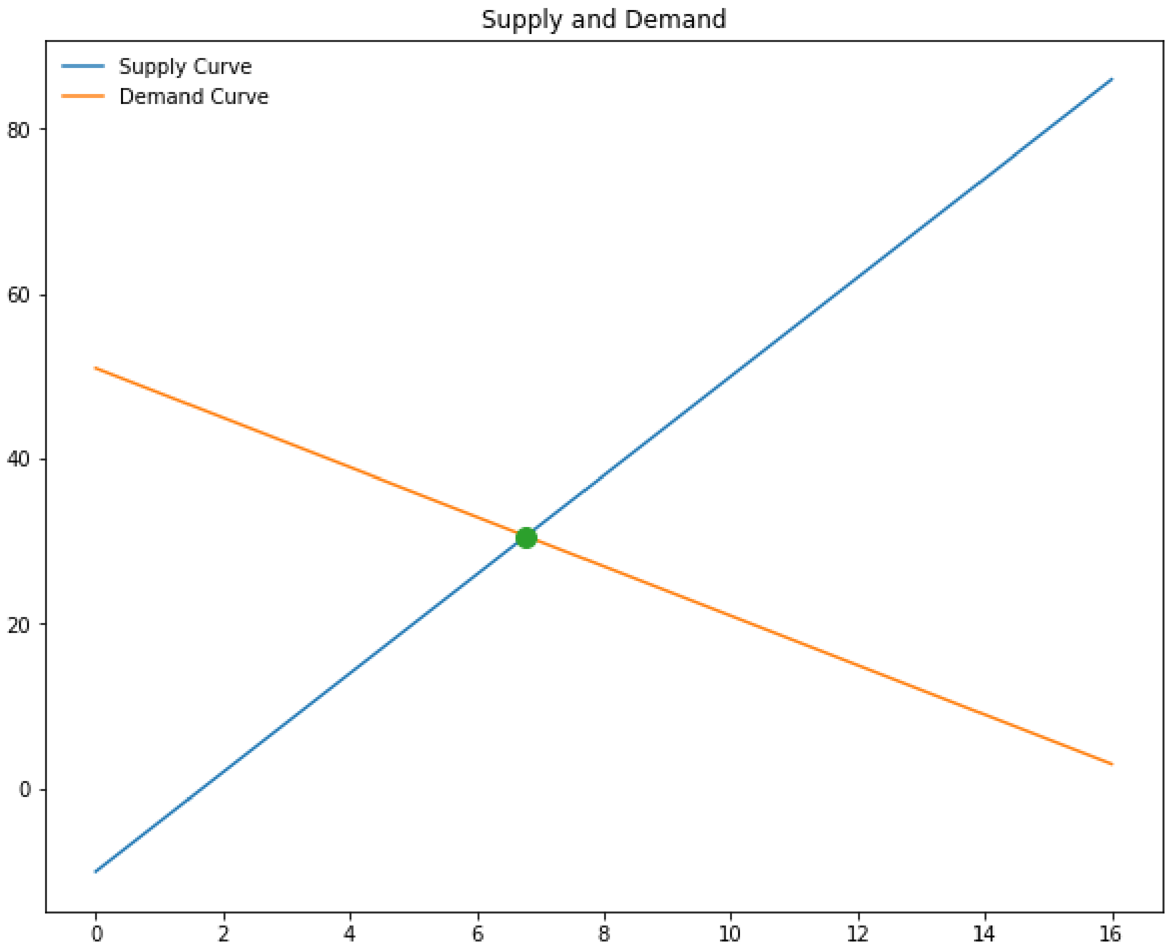
In [26]:
```
P = sy.Symbol('P')
eq = sy.Eq(S(P), D(P))
display(sy.solve(eq))
display(S(61/9))
```

```
[61/9]
30.666666666666664
```

By using https://calculus-notes.readthedocs.io/en/latest/0.8_consumer_surplus.html

In [27]:
```
plt.figure(figsize= (10, 8))
P = np.linspace(0, 16, 1000)


plt.plot(P, S(P), label = "Supply Curve")
plt.plot(P, D(P), label = "Demand Curve")
plt.plot(61/9, 30.666666666666664, 'o', markersize = 10)


plt.title("Supply and Demand")
plt.legend(frameon = False)



ax.annotate('Equilibrium at (61/9,30.666666666666664)',
    xy=(61/9,30.666666666666664),xytext=(61/9,30.666666666666664))
```

Out[27]: Text(6.777777777777778, 30.666666666666664, 'Equilibrium at (61/9,30.666666666666664)')



In [28]:
```
from sympy import symbols, Eq, solve
```

Solution by Elimination of Variables

In [10]:
```
P = sy.Symbol('P')
a = sy.Symbol('a')
b = sy.Symbol('b')
c = sy.Symbol('c')
d = sy.Symbol('d')
Q = sy.Symbol('Q')
eq1 = Eq(a + b*P,Q)
```

```python
eq2 = Eq(-c + d*P,Q)
solve((eq1,eq2), (P,Q))
```

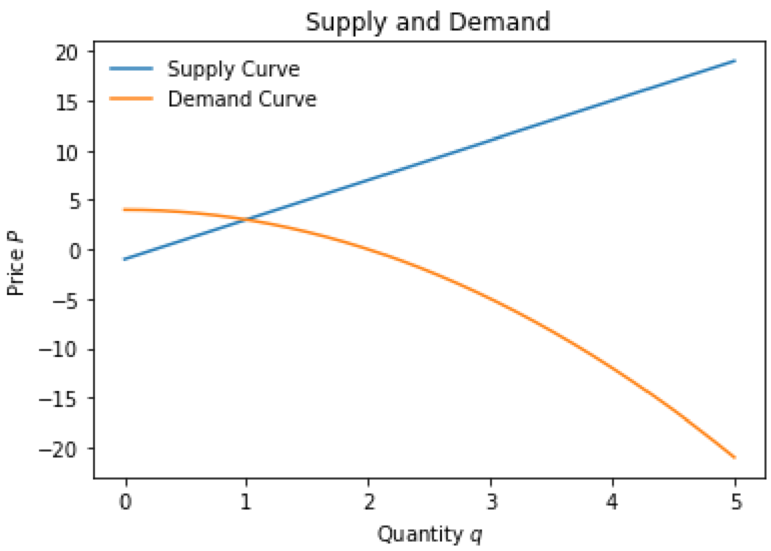
Out[10]: {Q: -(a*d + b*c)/(b - d), P: -(a + c)/(b - d)}

3.3 Partial Market Equilibrium—A Nonlinear ModeL

In [11]:
```python
def S(P,c=1,d=4):
    return (-c + d*P)

def D(P,a=4,b=1):
    return (a - b*P**2)

P = np.linspace(0, 5, 1000)
```

In [12]:
```python
plt.plot(P, S(P), label = "Supply Curve")
plt.plot(P, D(P), label = "Demand Curve")
plt.title("Supply and Demand")
plt.legend(frameon = False)
plt.xlabel("Quantity $q$")
plt.ylabel("Price $P$")
```

Out[12]: Text(0, 0.5, 'Price P')



In [13]:
```python
P = sy.Symbol('P')
```

```python
eq = sy.Eq(S(P), D(P))
display(sy.solve(eq))
display(S(1)) #We use 1 since price cannot be negative
```

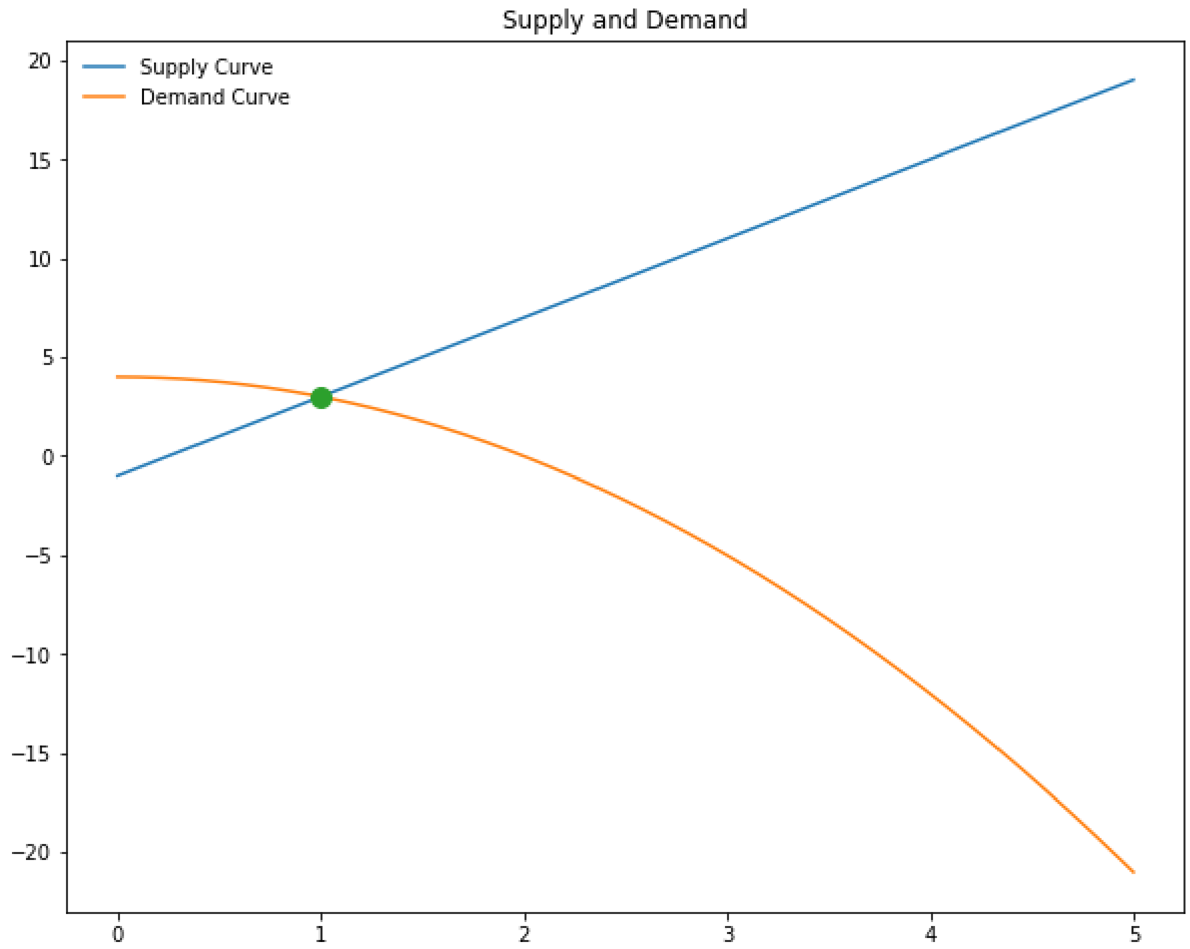
```
[-5, 1]
3
```

In [14]:

```python
plt.figure(figsize= (10, 8))
P = np.linspace(0, 5, 1000)


plt.plot(P, S(P), label = "Supply Curve")
plt.plot(P, D(P), label = "Demand Curve")
plt.plot(1,3, 'o', markersize = 10)


plt.title("Supply and Demand")
plt.legend(frameon = False)




ax.annotate('Equilibrium at (61/9,30.666666666666664)',
            xy=(1,3),xytext=(1,3 ),  arrowprops=dict(facecolor='black'))
```

Out[14]:

```
Text(1, 3, 'Equilibrium at (61/9,30.666666666666664)')
```



Higher-Degree Polynomial Equations --- Roots

In [15]:
```
from scipy import optimize
import matplotlib.pyplot as plt
```

In [16]:
```
def f(x):
    return (x**3 - x**2 - 4*x + 4)
x = np.array([-3, 0 , 3]) # Define an array which is near to  possible roots
roots = optimize.newton(f, x)
roots
```

Out[16]:
```
array([-2.,  1.,  2.])
```

3.4 General Market Equilibrium

In [17]:
```
P1 = sy.Symbol('P_1')
P2 = sy.Symbol('P_2')

c0 = sy.Symbol('c_0')
c1 = sy.Symbol('c_1')
c2 = sy.Symbol('c_2')
gamma0 = sy.Symbol('\\gamma_0')
gamma1 = sy.Symbol('\\gamma_1')
gamma2 = sy.Symbol('\\gamma_2')
eq1 = Eq(c1*P1 + c2*P2,-c0)
eq2 = Eq(gamma1*P1 + gamma2*P2,-gamma0)
display(eq1)
display(eq2)
```

$$P_1c_1 + P_2c_2 = -c_0$$

$$P_1\gamma_1 + P_2\gamma_2 = -\gamma_0$$

In [18]:
```
from sympy import symbols, Eq, solve
solve((eq1,eq2), (P1,P2))
```

Out[18]:
```
{P_1: (-\gamma_0*c_2 + \gamma_2*c_0)/(\gamma_1*c_2 - \gamma_2*c_1),
 P_2: (\gamma_0*c_1 - \gamma_1*c_0)/(\gamma_1*c_2 - \gamma_2*c_1)}
```

Numerical Example

In [19]:
```
eq1 = Eq(-5*P1 + 1*P2,-12)
eq2 = Eq(1*P1 +  -3*P2,-16)
display(eq1)
display(eq2)
solve((eq1,eq2), (P1,P2))
```

$$-5P_1 + P_2 = -12$$

$$P_1 - 3P_2 = -16$$

Out[19]:
```
{P_1: 26/7, P_2: 46/7}
```

3.5 Equilibrium in National-Income Analysis

In [20]:
```
Y = sy.Symbol('Y')
```

```
C = sy.Symbol('C')

I0 = sy.Symbol('I_0')
G0 = sy.Symbol('G_0')
a = sy.Symbol('a')
b = sy.Symbol('b')
eq1 = Eq(C + I0 + G0,Y)
eq2 = Eq(a + b *Y,C)
display(eq1)
display(eq2)
solve((eq1,eq2), (Y,C))
```

$$C + G_0 + I_0 = Y$$

$$Yb + a = C$$

Out[20]:
```
{C: -(a + b*(G_0 + I_0))/(b - 1), Y: -(G_0 + I_0 + a)/(b - 1)}
```

EXERCISE 3.5 Q3

In [21]:
```
eq1 = Eq(C + 16 + 14,Y)
eq2 = Eq(25 + 6*Y**(1/2),C)
display(eq1)
display(eq2)
solve((eq1,eq2), (Y,C))
```

$$C + 30 = Y$$

$$6Y^{0.5} + 25 = C$$

Out[21]:
```
[(121.000000000000, 91.0000000000000)]
```

Furkan Zengin