Multivariable Max Nov15 2016.txt-checkpoint

August 21, 2019

1 Multivariable Production Function

(2.23)

$$Y = h(X_1, X_2 | X_3, X_4)$$

(2.24)

$$Y = aX_1^{b_1} X_2^{b_2}$$

Let $a = 10, b_1 = 0.3, b_2 = 0.2$
Then

$$Y = 10 * X_1^{0.3} X_2^{0.2}$$

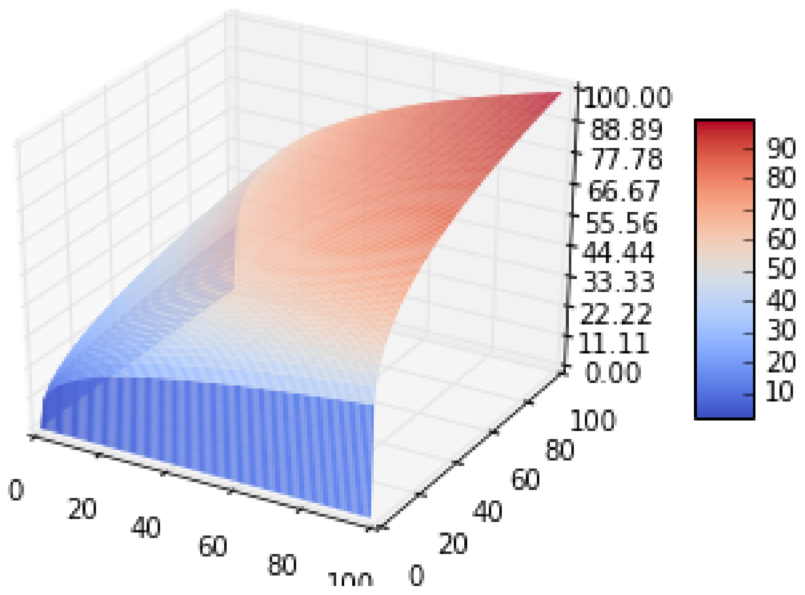
```
In [11]: %matplotlib inline
         ''' Plot the function '''
         from mpl_toolkits.mplot3d import Axes3D
         from matplotlib import cm
         from matplotlib.ticker import LinearLocator, FormatStrFormatter
         import matplotlib.pyplot as plt
         import numpy as np

         fig = plt.figure()
         ax = fig.gca(projection='3d')
         X = np.arange(0, 100, 1)
         Y = np.arange(0, 100, 1)
         X, Y = np.meshgrid(X, Y)
         Z = 10*(X**.3)*(Y**.2)
         surf = ax.plot_surface(X, Y, Z, rstride=1, cstride=1, cmap=cm.coolwarm,
                                linewidth=0, antialiased=True)
         ax.set_zlim(0, 100)

         ax.zaxis.set_major_locator(LinearLocator(10))
         ax.zaxis.set_major_formatter(FormatStrFormatter('%.02f'))

         fig.colorbar(surf, shrink=0.5, aspect=5)

         plt.show()
```



Economically optimum input use-levels

A profit function is given where P_y, P_1, and P_2 are the prices received and paid for the output and inputs, respectively, and FC represents fixed costs.

(2.30a)

$$\pi = P_y Y - P_1 X_1 - P_2 X_2 - FC$$

(2.30b)

$$\pi = P_y h(X_1, X_2 | X_3, X_4) - P_1 X_1 - P_2 X_2 - FC$$

Notice that the cost function is $C = P_1 X_1 + P_2 X_2 + FC$. Substituting the production function (2.23) for Y in (2.30), with both X_1 and X_2 variable, the profit equation becomes (2.30b). The assumed goal of the producer is profit maximization. Setting the first-order partial derivatives of (2.30b) with respect to X_1 and X_2 equal to zero, a system of two equations is obtained:

(2.31)

$$\frac{\partial \pi}{\partial X_1} = P_y \left(\frac{\partial Y}{\partial X_1} \right) - P_1 = 0 \Rightarrow \frac{\partial Y}{\partial X_1} = \frac{P_1}{P_y}$$

$$\frac{\partial \pi}{\partial X_2} = P_y \left(\frac{\partial Y}{\partial X_2} \right) - P_2 = 0 \Rightarrow \frac{\partial Y}{\partial X_2} = \frac{P_2}{P_y}$$

Solving this system simultaneously, we have the quantity each of X_1 and X_2 that will maximize profits relative to fixed resources.

Intuitively, the above equations mean that the value of marginal product of each input should equate their per unit cost of usage. That is, marginal revenue equates marginal cost at optimal choice of inputs. By 'optimal' we mean profit maximizing.

1.1 Employ the model in real world problem

Problem 1: 2 inputs 1 output

Consider a profit function for a farmer who uses water X_1 and fertilizer X_2 to produce a crop Y. The prices are P_1, P_2, and P_y, respectively. Assume there is no fixed cost, $FC = 0$. No factor of production can be negative. The farmer solves:

$$\max_{X_1,X_2,L} \pi = P_y Y - P_1 X_1 - P_2 X_2$$

$$\Rightarrow \max_{X_1,X_2} \pi = P_y(K X_1^{b_1} X_2^{b_2}) - P_1 X_1 - P_2 X_2$$

$$\text{s.t. } X_1 \geq 0, X_2 \geq 0$$

To find a solution using Python, use the following parameter values:

$$P_y = 20, P_1 = 1.35, P_2 = 1.50$$

$$k = 1, b_1 = 0.2, b_2 = 0.3$$

Find the optimal uses of water and fertilizer to produce the crop at optimal profit.

```
In [89]: """ Create environment """
         import numpy as np
         from scipy.optimize import minimize
         py=20.0
         p1=1.35
         p2=1.5
         b1=0.2
         b2=0.3
         k=1

         """ Define profit function """
         """ (we use negative sign before minimize command to maximize our function) """
         def func(x, sign=-1.0):
             return sign*(py*k*(x[0]**b1)*(x[1]**b2)-p1*x[0]-p2*x[1])
         """ Derivatives with respect to water and fertilizer """
         def func_deriv(x, sign=-1.0):
             dfdx0 = sign*(py*k*b1*(x[0]**(b1-1))*(x[1]**b2)-p1)
             dfdx1 = sign*(py*k*(x[0]**b1)*b2*(x[1]**(b2-1))-p2)
             return np.array([ dfdx0, dfdx1])
         """ Capacity constraints and their derivatives """
         cons = ({'type': 'ineq',
                  'fun' : lambda x: np.array([x[0]]),
                  'jac' : lambda x: np.array([1.0, 0.0])},
                 {'type': 'ineq',
                  'fun' : lambda x: np.array([x[1]]),
                  'jac' : lambda x: np.array([0.0, 1.0])})
         """Minimize the function with inequality constraints using SLSQP method"""
         res = minimize(func, [1.0,1.0], args=(-1.0,), jac=func_deriv,
                        constraints=cons, method='SLSQP', options={'disp': True})
         print("optimal water (X1) and fertilizer (X2)",np.around(res.x,decimals=2))
```

```
Optimization terminated successfully.    (Exit mode 0)
            Current function value: -35.4753332543
            Iterations: 12
            Function evaluations: 13
            Gradient evaluations: 12
optimal water (X1) and fertilizer (X2) [ 10.51  14.19]
```

Problem 2: 1 input 2 outputs

Now assume the farmer produces two crops Y_1 and Y_2. The farmer uses water X only and the prices are $P_1, P_2,$ and w_1, respectively. x cannot be negative.

Production function for crop Y_1 and Y_2 are, respectively:

$$Y_1 = X_1^a$$

$$Y_2 = X_1^b$$

Therefore the farmer solves the following problem:

$$\max_X \pi = P_1Y_1 + P_2Y_2 - wX$$

$$\Rightarrow \max_X \pi = P_1X^a + P_2X^b - wX$$

$$\text{s.t. } X \geq 0$$

To find a solution using Python, use the following parameter values:

$$P_1 = 20, P_2 = 30, w = 1.35$$

$$a = 0.2, b = 0.3$$

Find the uses of water and optimal output of each crop that maximize profit.

```
In [81]: """ Create environment """
         import numpy as np
         from scipy.optimize import minimize_scalar
         """ Set parameters """
         p1=20.0
         p2=30.0
         w=1.35
         a=0.2
         b=0.3
         infinity=1e99
         """ if we minimize negative of our function then it is equivalent to the maximize com
         f = lambda x: -(p1*(x**a)+p2*(x**b)-w*x)
         res = minimize_scalar(f, bounds=(0,infinity), method='bounded')
         print('water applied=',np.around(res.x,decimals=2))
         print('Y1=',np.around(res.x**a,decimals=2))
```

```
        print('Y2=',np.around(res.x**b,decimals=2))
        print('maximum profit=', np.around(p1*(res.x**a)+p2*(res.x**b)-w*res.x,decimals=2))
```

```
water applied= 22.48
Y1= 1.86
Y2= 2.54
maximum profit= 83.25
```

Problem 3: 2 inputs 2 outputs

We now proceed to more realistic scenario of multiple inputs and outputs. The farmer produces two crops Y_1 and Y_2. The farmer uses water X_1 and fertilizer X_2 and the prices are P_1, P_2, w_1 and w_2, respectively. The cost function has additive form $C = w_1 X_1 + w_2 X_2$. Water and fertilizer take only non-negative values.

Production function for crop Y_1 and Y_2 are, respectively:

$$Y_1 = X_1^{a_1} X_2^{a_2}$$

$$Y_2 = X_1^{b_1} X_2^{b_2}$$

Therefore the farmer solves the following problem:

$$\max_{X_1, X_2} \pi = P_1 Y_1 + P_2 Y_2 - w_1 X_1 - w_2 X_2$$

$$\Rightarrow \max_{X_1, X_2} \pi = P_1 (X_1^{a_1} X_2^{a_2}) + P_2 (X_1^{b_1} X_2^{b_2}) - w_1 X_1 - w_2 X_2$$

$$\text{s.t. } X_1 \geq 0, X_2 \geq 0$$

To find a solution using Python, use the following parameter values:

$$P_1 = 2.0, P_2 = 3.0, w_1 = 1.35, w_2 = 1.50$$

The production elasticities of water X_1 and fertilizer X_2 are:

$$\begin{bmatrix} a_1 & a_2 \\ b_1 & b_2 \end{bmatrix} = \begin{bmatrix} 0.2 & 0.3 \\ 0.4 & 0.5 \end{bmatrix}$$

Intuitively, a percent increase in X_1 increases production of Y_1 by $a_1 = 0.2$ percent and so on. Find the optimal uses of water and fertilizer to produce the crop at optimal profit.

```
In [117]: """ Create environment """
          import numpy as np
          from scipy.optimize import minimize
          p1=2.0
          p2=3.0
          w1=1.35
          w2=1.50
          a1=0.2
          a2=0.3
```

```python
b1=0.4
b2=0.5

""" Define profit function """
""" (we use negative sign before minimize command to maximize our function) """
def func(x, sign=-1.0):
    return sign*(p1*(x[0]**a1)*(x[1]**a2)+p2*(x[0]**b1)*(x[1]**b2)-w1*x[0]-w2*x[1])
""" Derivatives with respect to water and fertilizer """
def func_deriv(x, sign=-1.0):
    dfdx0 = sign*(a1*p1*x[0]**(a1-1)*x[1]**a2+b1*p2*x[0]**(b1-1)*x[1]**b2-w1)
    dfdx1 = sign*(a2*p1*x[0]**a1*x[1]**(a2-1)+b2*p2*x[0]**b1*x[1]**(b2-1)-w2)
    return np.array([ dfdx0, dfdx1])
""" Capacity constraints and their derivatives """
cons = ({'type': 'ineq',
         'fun' : lambda x: np.array([x[0]]),
         'jac' : lambda x: np.array([1.0, 0.0])},
        {'type': 'ineq',
         'fun' : lambda x: np.array([x[1]]),
         'jac' : lambda x: np.array([0.0, 1.0])})
"""Minimize the function with inequality constraints using SLSQP method"""
res = minimize(func, [1.0,1.0], args=(-1.0,), jac=func_deriv,
               constraints=cons, method='SLSQP', options={'disp': True})

print("optimal water (X1)=",np.around(res.x[0],decimals=2))
print("optimal fertilizer (X2)=",np.around(res.x[1],decimals=2))
print('one crop (Y1)=',np.around(res.x[0]**a1*res.x[1]**a2,decimals=2))
print('another crop (Y2)=',np.around(res.x[0]**b1*res.x[1]**b2,decimals=2))
print('maximum profit=', np.around( p1*(res.x[0]**a1)*(res.x[1]**a2)+p2*(res.x[0]**b
                                   *(res.x[1]**b2)-w1*res.x[0]-w2*res.x[1] ,decimals
```

```
Optimization terminated successfully.    (Exit mode 0)
            Current function value: -3.07827355945
            Iterations: 10
            Function evaluations: 10
            Gradient evaluations: 10
optimal water (X1)= 3.73
optimal fertilizer (X2)= 4.34
one crop (Y1)= 2.02
another crop (Y2)= 3.53
maximum profit= 3.08
```

Problem 4: 2 inputs 2 outputs (introducing land constraint)

So far we considered land unlimited. The following problem assumes the same specifications as in Problem(3), but it includes land as factor of production. Instead of fertilizer, let X_2 be the amount of land used for production. Total land available to the farmer is $\bar{L}$. Notice that if $X_2 = 0$ then no production is possible. So we must set $0 < X_2 \leq \bar{L}$

Production function for crop Y_1 and Y_2 are, respectively:

$$Y_1 = X_1^{a_1} X_2^{a_2}$$

$$Y_2 = X_1^{b_1} X_2^{b_2}$$

Therefore the farmer solves the following problem:

$$\max_{X_1, X_2} \pi = P_1 Y_1 + P_2 Y_2 - w_1 X_1 - w_2 X_2$$

$$\Rightarrow \max_{X_1, X_2} \pi = P_1(X_1^{a_1} X_2^{a_2}) + P_2(X_1^{b_1} X_2^{b_2}) - w_1 X_1 - w_2 X_2$$

$$\text{s.t. } X_1 \geq 0$$

$$0 < X_2 \leq \bar{L} \Rightarrow \bar{L} - X_2 \geq 0$$

To find a solution using Python, use the same parameter values:

$$P_1 = 2.0, P_2 = 3.0, w_1 = 1.35, w_2 = 1.50, \bar{L} = 4$$

Let the production elasticities of water X_1 and land X_2 be:

$$\begin{bmatrix} a_1 & a_2 \\ b_1 & b_2 \end{bmatrix} = \begin{bmatrix} 0.2 & 0.3 \\ 0.4 & 0.5 \end{bmatrix}$$

Intuitively, a percent increase in X_1 increases production of Y_1 by $a_1 = 0.2$ percent and so on. Find the optimal uses of water and fertilizer to produce the crop at optimal profit.

```
In [3]: """ Create environment """
        import numpy as np
        from scipy.optimize import minimize
        p1=2.0
        p2=3.0
        w1=1.35
        w2=1.50
        a1=0.2
        a2=0.3
        b1=0.4
        b2=0.5
        lbar=4.0

        """ Define profit function """
        """ (we use negative sign before minimize command to maximize our function) """
        def func(x, sign=-1.0):
            return sign*(p1*(x[0]**a1)*(x[1]**a2)+p2*(x[0]**b1)*(x[1]**b2)-w1*x[0]-w2*x[1])
        """ Derivatives with respect to water and fertilizer """
        def func_deriv(x, sign=-1.0):
            dfdx0 = sign*(a1*p1*x[0]**(a1-1)*x[1]**a2+b1*p2*x[0]**(b1-1)*x[1]**b2-w1)
            dfdx1 = sign*(a2*p1*x[0]**a1*x[1]**(a2-1)+b2*p2*x[0]**b1*x[1]**(b2-1)-w2)
```