In [27]:
```
def generation(lam,fc,limits,n,b):
    for i in range(n):
        x = (lam - fc[i,1])/(2*(fc[i,0] + lam*b[i]))
        pgen.append(x)
        if(pgen[i]<limits[i,0]):
            pgen[i] = limits[i,0]
        if(pgen[i]>limits[i,1]):
            pgen[i] = limits[i,1]


    print(pgen)
```

In [3]:
```
pd = float(input('Enter the value of pd:'))
n = int(input('Enter numbers of generators:'))
rows = n
cols = 3
```

```
Enter the value of pd:412.35
Enter numbers of generators:2
```

In [4]:
```
import numpy as np
```

In [5]:
```
fc= np.zeros((rows,cols))
for i in range(rows):
    for j in range(cols):
        value = float(input('Enter parameters of fuel cost:'))
        fc[i,j] = value
np.set_printoptions(suppress = True)
print(fc)
```

```
Enter parameters of fuel cost:0.004
Enter parameters of fuel cost:6.2
Enter parameters of fuel cost:320
Enter parameters of fuel cost:0.003
Enter parameters of fuel cost:6
Enter parameters of fuel cost:200
[[  0.004   6.2   320.   ]
 [  0.003   6.    200.   ]]
```

In [6]:
```
limits= np.zeros((rows,2))
for i in range(rows):
    for j in range(2):
        value = float(input('Enter lower limit and upper limit:'))
        limits[i,j] = value
np.set_printoptions(suppress = True)
print(limits)
```

```
Enter lower limit and upper limit:50
Enter lower limit and upper limit:250
Enter lower limit and upper limit:50
Enter lower limit and upper limit:250
[[ 50. 250.]
 [ 50. 250.]]
```

In [7]:
```
b = []
for i in range(n):
    x = float(input('Enter loss coefficients:'))
    b.append(x)
print(b)
```

```
Enter loss coefficients:0.00000125
Enter loss coefficients:0.000000625
[1.25e-06, 6.25e-07]
```

In [17]:
```
lam = np.max(fc[:,1])
lam
```

Out[17]:
```
6.2
```

In [29]:
```
pgen = []
fgen = []
```

In [30]:
```
generation(lam,fc,limits,n,b)
```

```
[50.0, 50.0]
```

In [32]:
```
pl = 0
for i in range(n):
    pl = pl + b[i]*(pgen[i])**2
delp = pd + pl - sum(pgen)
delp
```

Out[32]:
```
312.3546875
```

In [33]:
```
while(abs(delp)>0.0032):
    den = 0
    for i in range(n):
        den = den + (fc[i,0]+fc[i,1]*b[i])/(2*((fc[i,0] + lam*b[i])**2))
    del_lambda = delp/den
    lam = lam + del_lambda
    pgen.clear()
    generation(lam,fc,limits,n,b)
    pl = 0
    for i in range(n):
        pl = pl + b[i]*(pgen[i])**2
    delp = pd + pl - sum(pgen)
```

```
[133.7754018641315, 211.78502835127907]
[162.40553667334558, 249.9909909770098]
[162.41646042470714, 250.0]
[162.418848928438, 250.0]
```

In [34]:
```
for i in range(n):
    x = (lam - fc[i,1])/(2*(fc[i,0]))
    pgen.append(x)
    print('unit {} generation = {} MW\n'.format(i,pgen[i]))

print('---------------------------------------------------------------------------------------------')
print('incrimental cost = {} $/H'.format(lam))
print('---------------------------------------------------------------------------------------------')
for i in range(n):
    y = fc[i,0]*pgen[i]**2 + fc[i,1]*pgen[i] + fc[i,2]
    fgen.append(y)
    print('unit {} cost = {} $/H\n'.format(i,fgen[i]))
print('---------------------------------------------------------------------------------------------')

ft = sum(fgen)
print('total cost = {} $/H\n'.format(ft))
```

```
unit 0 generation = 162.418848928438 MW

unit 1 generation = 250.0 MW

---------------------------------------------------------------------------------------------
incrimental cost = 7.502397118187854 $/H
---------------------------------------------------------------------------------------------
unit 0 cost = 1432.5163933052706 $/H

unit 1 cost = 1887.5 $/H

---------------------------------------------------------------------------------------------
total cost = 3320.016393305271 $/H
```

In [ ]:

In [ ]: