Example 3.3. A manufacturer of lawn furniture makes two types of lawn chairs, one with a wood frame and one with a tubular aluminum frame. The wood–frame model costs \$20 per unit to manufacture, and the aluminum–frame model costs \$12 per unit. The company operates in a market where the number of units that can be sold depends on the price. It is estimated that in order to sell x units per day of the wood–frame model and y units per day of the aluminum–frame model, the selling price cannot exceed $10 + 31x^{-0.5} + 1.3y^{-0.2}$ \$/unit for wood–frame chairs, and $5 + 15y^{-0.4} + 0.8x^{-0.08}$ \$/unit for aluminum– frame chairs. Find the optimal production levels.

$x_1 =$ the number of wood-frame chairs
$x_2 =$ the number of aluminum-frame chairs
$p_1(x) =$ price to sell x wood-frame chairs
$p_2(x) =$ price to sell y wood-frame chairs
$P(x) =$ profit

In [1]:
```
using Symbolics
D(f,x)=expand_derivatives(Differential(x)(f))
@variables x,y
```

Out[1]:
```
2-element Vector{Num}:
 x
 y
```

In [2]:
```
p1(x,y)=10+31*x^(-0.5)+1.3*y^(-0.2)
p2(x,y)=5+15*y^(-0.4)+0.8*x^(-0.08)
P(x,y)=(p1(x,y)-20)*x+(p2(x,y)-12)*y
P(x,y)
```

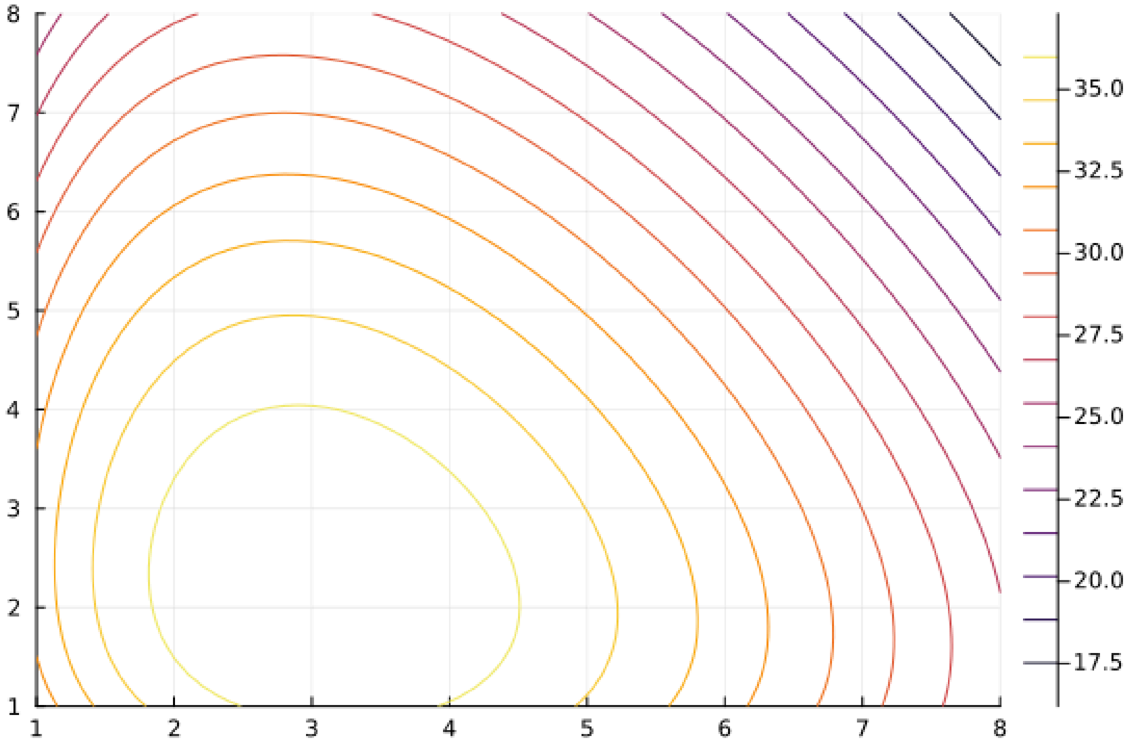
Out[2]:
```
x*(-10 + 1.3 / (y^0.2) + 31 / (x^0.5)) + (-7 + 0.8 / (x^0.08) + 15 / (y^0.
4))*y
```

(i) Graph level curves of the profit $P(x, y)$ where x is the number of wood-frame chairs produced and y is the number of aluminum-frame chairs for $x \in [1, 8]$ and $y \in [1, 8]$.

In [3]:
```
using Plots
```

In [4]:
```
contour(1:0.1:8,1:0.1:8,P)
```

Out[4]:



(ii) Use Newton’s method for two variables to find the optimal pro duction levels and the corresponding profit.

In [5]:

```
gradP=Symbolics.gradient(P(x,y),[x,y])
gradPs="F(x,y)="*string(Symbolics.toexpr(gradP))
eval(Meta.parse(gradPs))
F(x,y)
```

Out[5]:

```
2-element Vector{Num}:
                             -10 + 1.3 / (y^0.2) + -15.5 / (x^0.5) + 31 / (x
^0.5) + (-0.064y) / (x^1.08)
 -7 + 0.8 / (x^0.08) + -6.0 / (y^0.3999999999999999) + 15 / (y^0.4) + (-0.2
6x) / (y^1.2000000000000002)
```

In [6]:

```
DFn=Symbolics.jacobian(F(x,y),[x,y])
DFs="DF(x,y)="*string(Symbolics.toexpr(DFn))
eval(Meta.parse(DFs))
DF(y,x)
```

Out[6]:

```
2×2 Matrix{Num}:
 (0.06912x) / (y^2.08) + 7.75 / (y^1.5) + -15.5 / (y^1.5)  …
-0.064 / (y^1.08) + -0.26 / (x^1.2)
                      -0.064 / (y^1.08) + -0.26 / (x^1.2)      (0.312y) / (x
^2.2) + 2.4 / (x^1.4) + -6.0 / (x^1.4)
```

In [7]:

```
Xn=[3,2]
for n=1:5
    Xn=Xn-DF(Xn...)\F(Xn...)
    println("X_$n=",Xn)
end
Xopt=Xn
```

```
Popt=P(Xopt...)
println("\nThe optimal production levels are")
println("\tWood-frame chairs: ",Xopt[1])
println("\tAluminum-fram chairs: ",Xopt[2])
println("The optimal profit is ",Popt)
```

```
X_1=[3.0116984973338825, 2.1829892780771383]
X_2=[3.01154242501657, 2.1941974399412607]
X_3=[3.01154179832089, 2.1942340630131656]
X_4=[3.0115417983142767, 2.194234063401063]
X_5=[3.011541798314277, 2.194234063401064]

The optimal production levels are
        Wood-frame chairs: 3.011541798314277
        Aluminum-fram chairs: 2.194234063401064
The optimal profit is 37.31045402768908
```

(iii) Denote by c the per-unit cost of the aluminum-frame chairs and compute $S(x, c)$, $S(y, c)$ and $S(P, c)$ evaluated at $c = 12$. Here P is the profit at the optimal production values x and y.

In [8]:
```
@variables c
```

Out[8]:
```
1-element Vector{Num}:
 c
```

In [9]:
```
P(x,y)=(p1(x,y)-20)*x+(p2(x,y)-c)*y
gradP=Symbolics.gradient(P(x,y),[x,y])
gradPs="F(x,y)="*string(Symbolics.toexpr(gradP))
eval(Meta.parse(gradPs))
DFn=Symbolics.jacobian(F(x,y),[x,y])
DFs="DF(x,y)="*string(Symbolics.toexpr(DFn))
eval(Meta.parse(DFs))
```

Out[9]:
```
DF (generic function with 1 method)
```

At the optimum we have $F(x, y) = 0$ where $F = \nabla P$. Write $F = (f_1, f_2)$ and use implicit differetiation to obtain

$$\frac{df_1}{dc} = \frac{\partial f_1}{\partial x}\frac{dx}{dc} + \frac{\partial f_1}{\partial y}\frac{dy}{dc} + \frac{\partial f_1}{\partial c} = 0$$

$$\frac{df_2}{dc} = \frac{\partial f_2}{\partial x}\frac{dx}{dc} + \frac{\partial f_2}{\partial y}\frac{dy}{dc} + \frac{\partial f_2}{\partial c} = 0$$

In vector format this may be written as

$$\begin{bmatrix} \partial f_1/\partial x & \partial f_1/\partial y \\ \partial f_2/\partial x & \partial f_2/\partial y \end{bmatrix} \begin{bmatrix} dx/dc \\ dy/dc \end{bmatrix} = - \begin{bmatrix} \partial f_1/\partial c \\ \partial f_2/\partial c \end{bmatrix}.$$

In [10]:
```
F(x,y)
```

Out[10]: 2-element Vector{Num}:

```
                              -10 + 1.3 / (y^0.2) + -15.5 / (x^0.5) + 31 / (x^0.5) + (-0.064y) / (x^1.08)
 5 + 0.8 / (x^0.08) + -6.0 / (y^0.3999999999999999) + 15 / (y^0.4) - c + (-0.26x) / (y^1.2000000000000002)
```

In [11]:

```
# Convenient definition to evaluate things
# at (x,y)=(xopt,yopt) and c=12
vopt=[c=>12,x=>Xopt[1],y=>Xopt[2]]
```

Out[11]:

```
3-element Vector{Pair{Num, Float64}}:
 c => 12.0
 x => 3.011541798314277
 y => 2.194234063401064
```

For computational efficiency substitute $c = 12$ and the corresponding optimal production values $(x, y) = (x_{\text{opt}}, y_{\text{opt}})$ before solving the matrix equation.

In [12]:

```
# Take partial derivatives element by element
pFpc=(z->D(z,c)).(F(x,y))
# Evaluate at c=12 and corresponding production levels
tmp=substitute(pFpc,vopt)
pFpcopt=eval(Symbolics.toexpr(tmp))
```

Out[12]:

```
2-element Vector{Int64}:
  0
 -1
```

In [13]:

```
tmp=substitute(DF(x,y),vopt)
# Evaluate at c=12 and corresponding production levels
DFopt=eval(Symbolics.toexpr(tmp))
```

Out[13]:

```
2×2 Matrix{Float64}:
 -1.46761   -0.120717
 -0.120717  -1.03137
```

Solve the system $(DF)(dX/dc) = -\partial F/\partial c$.

In [14]:

```
dXdc=-DFopt\pFpcopt
```

Out[14]:

```
2-element Vector{Float64}:
  0.08052759557880806
 -0.9790137492179721
```

In [15]:

```
Sxc=12/Xopt[1]*dXdc[1]
println("S(x,c)=",Sxc)
```

```
S(x,c)=0.32087588738984285
```

In [16]:

```
Syc=12/Xopt[2]*dXdc[2]
println("S(y,c)=",Syc)
```

```
S(y,c)=-5.354107470378982
```

By the chain rule

$$\frac{dP}{dc} = \nabla P \frac{dX}{dc} + \frac{\partial P}{\partial c}.$$

At the optimal production levels $\nabla P = 0$, therefore

$$S(P,c) = \frac{c}{P}\frac{\partial P}{\partial c}\bigg|_{c=12, x=x_{\text{opt}}, y=y_{\text{opt}}}$$

In [17]:
```
pPpc=D(P(x,y),c)
tmp=substitute(pPpc,vopt)
pPpcopt=eval(Symbolics.toexpr(tmp))
```

Out[17]:
```
-2.194234063401064
```

In [18]:
```
SPc=12/Popt*pPpcopt
println("S(P,c)=",SPc)
```

```
S(P,c)=-0.7057220140304906
```

(iv) Explain intuitively why each of the sensitivites $S(x,c)$, $S(y,c)$ and $S(P,c)$ are respectively either positive or negative.

If the cost of producing aluminum-frame chairs goes up then the optimal production levels should switch to more wooden-frame chairs and less aluminum-frame chairs.

Since $S(x,c) > 0$ that means more wooden-frame chairs.

Since $S(y,c) < 0$ that means less aluminum-frame chairs.

When production costs go up when c increases we expect the profit to go down. This is why $S(P,c) < 0$.

In [ ]:
```

```