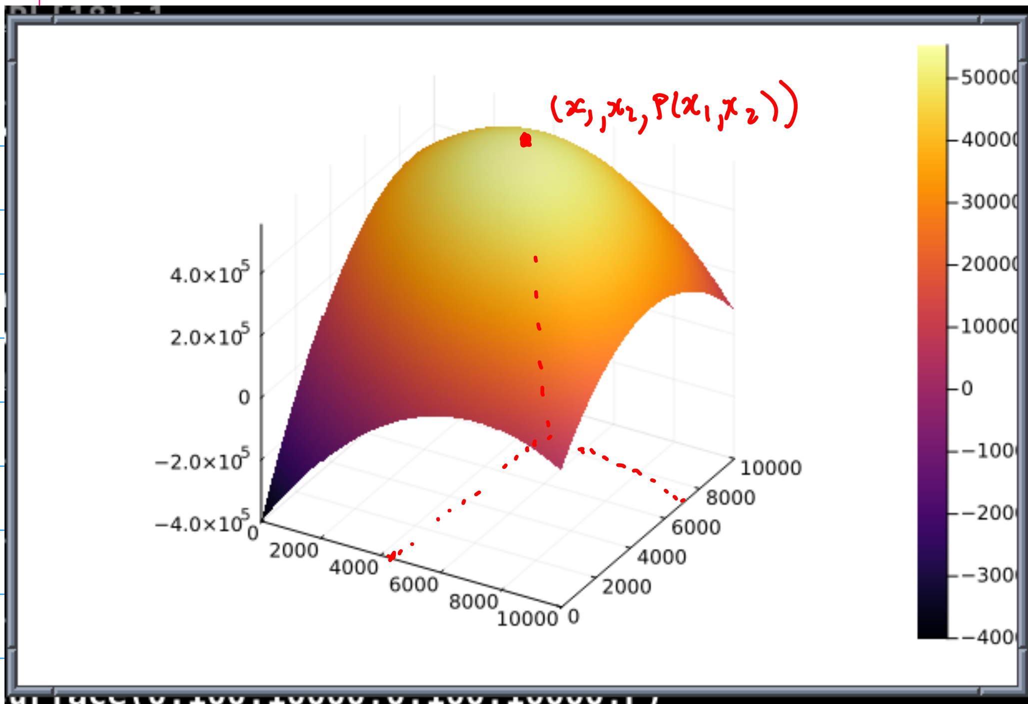
From last time:

```
julia> p1(x1,x2)=339-1//100*x1-3//1000*x2
p1 (generic function with 1 method)

julia> p2(x1,x2)=399-4//1000*x1-1//100*x2
p2 (generic function with 1 method)

julia> C(x1,x2)=400_000+195*x1+225*x2
C (generic function with 1 method)

julia> R(x1,x2)=x1*p1(x1,x2)+x2*p2(x1,x2)
R (generic function with 1 method)

julia> P(x1,x2)=R(x1,x2)-C(x1,x2)
P (generic function with 1 method)
```

```
julia> using Symbolics

julia> @variables x1,x2
2-element Vector{Num}:
 x1
 x2

julia> P(x1,x2)
-400000 - 195x1 - 225x2 + x1*(339 - (1//100)*x1 - (3//1000)*x2) + (399 - (1//250
)*x1 - (1//100)*x2)*x2

julia> D(f,x)=expand_derivatives(Differential(x)(f))
D (generic function with 1 method)

julia> dpdx1=D(P(x1,x2),x1)
144 - (1//50)*x1 - (7//1000)*x2

julia> dpdx2=D(P(x1,x2),x2)
174 - (7//1000)*x1 - (1//50)*x2
```

```
julia> A=[1//50 7//1000; 7//1000 1//50]
2×2 Matrix{Rational{Int64}}:
 1//50    7//1000
 7//1000  1//50

julia> b=[144,174]
2-element Vector{Int64}:
 144
 174

julia> x=A\b
2-element Vector{Rational{Int64}}:
 554000//117
 824000//117
```

Maximum...




```
julia> Float64.([ x[1],x[2],P(x[1],x[2])])
3-element Vector{Float64}:
   4735.042735042735
   7042.735042735043
 553641.0256410256
```

Sensitivity:

Example 2.1. A manufacturer of color TV sets is planning the introduction of two new products, a 19–inch LCD flat panel set with a manufacturer's suggested retail price (MSRP) of \$339 and a 21–inch LCD flat panel set with an MSRP of \$399. The cost to the company is \$195 per 19–inch set and \$225 per 21–inch set, plus an additional \$400,000 in fixed costs. In the competitive market in which these sets will be sold, the number of sales per year will affect the average selling price. It is estimated that for each type of set, the average selling price drops by one cent for each additional unit sold. Furthermore, sales of the 19–inch set will affect sales of the 21–inch set, and vice–versa. It is estimated that the average selling price for the 19–inch set will be reduced by an additional 0.3 cents for each 21–inch set sold, and the price for the 21–inch set will decrease by 0.4 cents for each 19–inch set sold. How many units of each type of set should be manufactured?

What if the elasticity of 19" televisions was a rather than 0.01?

Find optimal x_1 and x_2 as functions of a, and then find

$$S(x_1,a) = \frac{a}{x_1}\frac{dx_1}{da} \qquad S(x_2,a) = \frac{a}{x_2}\frac{dx_2}{da}$$

```
julia> p1(x1,x2)=339-a*x1-3//1000*x2
p1 (generic function with 1 method)

julia> @variables a
1-element Vector{Num}:
 a

julia> dpdx1=D(P(x1,x2),x1)
144 - (7//1000)*x2 - 2a*x1

julia> dpdx2=D(P(x1,x2),x2)
174 - (7//1000)*x1 - (1//50)*x2

julia> A=[2*a 7//1000; 7//1000 1//50]
2×2 Matrix{Num}:
      2a   7//1000
 7//1000     1//50

julia> xa=A\b
2-element Vector{Num}:
 (174 + ((-1//50)*(144 - (348000//7)*a)) / ((7//1000) - (40//7)*a)) / (7//1000)
                                  (144 - (348000//7)*a) / ((7//1000) - (40//7)*a)
```

$$x_1 = \frac{1,662,000}{40,000a - 49}$$

$$x_2 = 8,700 - \frac{581,700}{40,000a - 49}.$$

```
julia> simplify(xa[1])
(174((7//1000) - (40//7)*a) - (1//50)*(144 - (348000//7)*a)) / ((7//1000)*((7//1
000) - (40//7)*a))
```

Remark: the built-in simplication routine made it "worse"...

```
julia> expand(xa[1])
(174000//7) + ((1000//7)*(-(72//25) + (6960//7)*a)) / ((7//1000) - (40//7)*a)
```

A little bit better, but still not "simple"

$$S(x_1,a) = \frac{a}{x_1}\frac{dx_1}{da}\bigg|_{\substack{a=.01\\ x_1=\text{opt for } a=.01}}$$

$$\frac{dx_1}{da} = \frac{-66,480,000,000}{(40,000a - 49)^2}$$

```
julia> D(xa[1],a)
(((-4//35)*(144 - (348000//7)*a)) / (((7//1000) - (40//7)*a)^2) + (6960//7) / ((
7//1000) - (40//7)*a)) / (7//1000)

julia> simplify(D(xa[1],a))
(831//612500) / (-(49//1000000000000) + (1//12500000)*a - (1//30625)*(a^2))
```

```
julia> dx1da=simplify(D(xa[1],a))
(831//612500) / (-(49//1000000000000) + (1//12500000)*a - (1//30625)*(a^2))

julia> substitute(a/x[1]*dx1da,a=>0.01)
-1.1396011396011398
```

```
julia> -400/351
-1.1396011396011396
```

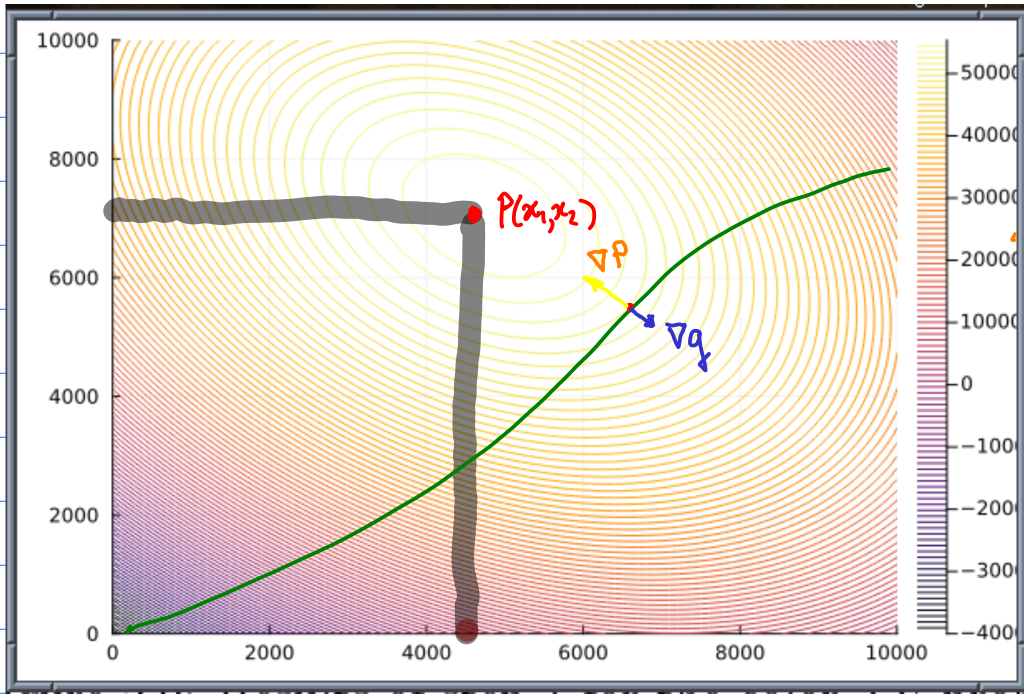
Same answer as book...

```
julia> dx2da=simplify(D(xa[2],a))
(831//12250) / ((7//1000000000) - (1//87500)*a + (8//1715)*(a^2))

julia> substitute(a/x[2]*dx2da,a=>0.01)
0.26816585069012255
```

$$S(x_2, a) = \frac{9,695}{36,153} \approx 0.27.$$

Example 2.2. We reconsider the color TV problem (Example 2.1) introduced in the previous section. There we assumed that the company has the potential to produce any number of TV sets per year. Now we will introduce constraints based on the available production capacity. Consideration of these two new products came about because the company plans to discontinue manufacture of some older models, thus providing excess capacity at its assembly plant. This excess capacity could be used to increase production of other existing product lines, but the company feels that the new products will be more profitable. It is estimated that the available production capacity will be sufficient to produce 10,000 sets per year ($\approx$ 200 per week). The company has an ample supply of 19–inch and 21–inch LCD panels and other standard components; however, the circuit boards necessary for constructing the sets are currently in short supply. Also, the 19–inch TV requires a different board than the 21–inch model because of the internal configuration, which cannot be changed without a major redesign, which the company is not prepared to undertake at this time. The supplier is able to supply 8,000 boards per year for the 21–inch model and 5,000 boards per year for the 19–inch model. Taking this information into account, how should the company set production levels?

Review of the idea of Lagrange multipliers...



Find maximum on the line

$g(x_1, x_2) = c$

constraint

$\nabla P = \lambda \nabla g$

Lagrange multiplier...

```
julia> Float64.([ x[1],x[2],P(x[1],x[2])])
3-element Vector{Float64}:
   4735.042735042735
   7042.735042735043
 553641.0256410256
```

page 34

Example 2.3. Maximize $x + 2y + 3z$ over the set $\underbrace{x^2 + y^2 + z^2 = 3}_{g(x,y,z)=3 \text{ constraint}}$.

read this!