

Solving Linear Programs

Solving (non-)Linear Programs

Three goals for this assignment:

1. Develop intuition about shadow prices.
2. Get repetitions of **optimization set-up**
3. Solidify your **optimization workflow**

If you can walk someone through how to do these optimizations using your tool of choice then you are making good progress.

Shadow prices

Let's think about cost, and how cost depends on the way a choice problem is framed. Let's imagine that your company now produces two products. Let x and y , respectively, denote the quantities of the two products that are produced and sold. Any nonnegative quantities satisfying the following constraints can be produced:

$$\begin{aligned}x + y &\leq 400 \\x + 2y &\leq 500\end{aligned}$$

Your firm's revenue is given by $40x + 42y$, and your cost is given by $30x + 30y$. (Though this is clearly a short-run story, as capacity is fixed, we suppress the fixed cost and measure "profit" as $10x + 12y$.)

Do the following:

1. Determine an optimal solution for your firm.
2. Determine the shadow prices on the two constraints.
3. In what sense are the shadow prices on the two constraints opportunity costs?

Step 1: Choice variables

What are we choosing in this framing of the problem?

x and y . Include any natural (implied) constraints, i.e. no negative production.

```
# fire up gekko
from gekko import GEKKO
m=GEKKO(remote=False)
# define the choices
x=m.Var(name="x",lb=0)
x.value=1
y=m.Var(name="y",lb=0)
y.value=1
```

Step 2: Objective Function

We want to **maximize** profit.

$$\Pi = 10x + 12y$$

```
# and write down the objective function
m.Maximize(10*x+12*y) # version for gekko
def profit(x,y):
    return 10*x+12*y # version for use to use later
```

Step 3: Constraints

$$x + y \leq 400$$

$$x + 2y \leq 500$$

You may simplify here if needed. Once you've specified everything your set up is complete.

```
m.Equation(x+y<400)
m.Equation(x+2*y<500)
```

```
<gekko.gekko.EquationObj at 0x166065ac0>
```

Step 4: SOLVE

```
m.solve(disp=False)
pi_0=profit(x.value[0],y.value[0])
print("x:      ", int(x.value[0]))
print("y:      ", int(y.value[0]))
print("Profit: ",pi_0)
```

```
x:      300
y:      100
Profit: 4200.0
```

What the heck is a shadow price?

For a given objective function and constraint, the *shadow price* on the constraint is the **rate at which the value of the objective function changes as the constraint is relaxed**.

- Notate bene: Shadow prices are often called Lagrange multipliers.

What the heck is a shadow price?

We often are interested in calculating the change over a specific number of units, as this allows us to characterize the benefit of relaxing the constraint by a specific increment.

You can think of the *shadow price* as the opportunity cost of **not** relaxing the constraint.

Shadow prices are key when we consider the costs and benefits of relaxing constraints.

So lets relax the first constraint by 1 unit

```
# same set up as before
from gekko import GEKKO
m=GEKKO(remote=False)
# define the variables
x=m.Var(name="x",lb=0)
x.value=1
y=m.Var(name="y",lb=0)
y.value=1
# EQUATIONS
```

```

m.Maximize(10*x+12*y)
m.Equation(x+y<401) # <-- added 1, no other changes
m.Equation(x+2*y<500)

```

<gekko.gekko.EquationObj at 0x166065ca0>

So lets relax the first constraint by 1 unit

Result:

```

m.solve(disp=False)
pi_1=profit(x.value[0],y.value[0])
print("x:      ", x.value[0])
print("y:      ", y.value[0])
print("Profit: ",pi_1)
print("Benefit of relaxing the first constraint by one unit: ",int(pi_1-pi_0))

```

```

x:      302.0
y:      99.0
Profit: 4208.0
Benefit of relaxing the first constraint by one unit:  8

```

Then lets relax the second constraint by 1 unit

```

# same set up as before
from gekko import GEKKO
m=GEKKO(remote=False)
# define the variables
x=m.Var(name="x",lb=0)
x.value=1
y=m.Var(name="y",lb=0)
y.value=1
# EQUATIONS
m.Maximize(10*x+12*y)
m.Equation(x+y<400) # back to the original value
m.Equation(x+2*y<501) # <-- added 1, no other changes

```

<gekko.gekko.EquationObj at 0x166065ee0>

Then lets relax the second constraint by 1 unit

Result:

```
m.solve(disp=False)
pi_2=profit(x.value[0],y.value[0])
print("x:      ", x.value[0])
print("y:      ", y.value[0])
print("Profit: ",pi_2)
print("Benefit of relaxing the second constraint by one unit: ",int(pi_2-pi_0))
```

```
x:      299.0
y:      101.0
Profit: 4202.0
Benefit of relaxing the second constraint by one unit:  2
```

Component Searches and Product Cost

Let's keep thinking about the scenario above. Now let's think in terms of how many units of the first product, x , to produce and sell. Clearly we require $0 \leq x \leq 400$. Within this range, it should also be clear you would produce as many units of the second product as possible. The constraints imply, for any such x , the corresponding choice of y would be $y = g(x) = \min\{400 - x, 0.5(500 - x)\}$.

This implies profit as a function of x given by $10x + 12g(x) = 10x + 12 \times \min\{400 - x, 0.5(500 - x)\}$.

4.1 Plot the profit function

$$10x + 12g(x) = 10x + 12 \times \min\{400 - x, 0.5(500 - x)\}.$$

Let's set this up as simply as possible

```
# anytime we want to plot np and pd will be a great place to start
import numpy as np
import pandas as pd
# set up the x range
minx=0
maxx=400
x=np.linspace(minx,maxx,maxx)
```

Keep things simple

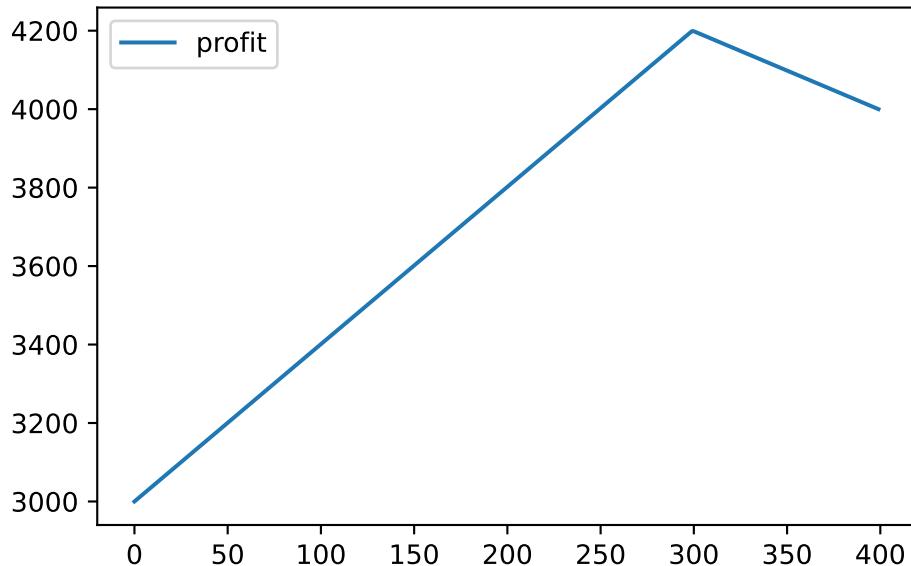
$$\Pi = 10x + 12g(x) = 10x + 12 \times \min\{400 - x, 0.5(500 - x)\}$$

We just have two options for $g(x)$, calculate both, then apply the condition.

```
# calc both g(x) options
g_0=400-x
g_1=.5*(500-x)
# calc y using the np.where function
y = np.where(g_0<=g_1,g_0,g_1)
# and calc profit
profit=10*x+12*y
```

At this point we can toss this to pandas and plot

```
df=pd.DataFrame({"profit":profit})
df.plot()
```



4.2 The optimal point is at $x = 300$ which implies $y = 100$ in this framing

5. Next, observe (but verify that) this function simplifies to $10x + 3,000 - 6x$ if $0 \leq x \leq 300$ and $10x + 4800 - 12x$ if $300 \leq x \leq 400$. Concentrate on the first range. What is the

implied incremental or marginal cost of the first product in this range? Carefully explain your answer, in light of the fact this product was previously viewed as providing revenue of 40 per unit less cost of 30 per unit.

6. Why does the cost of the product depend on the decision frame?

Why is the profit function $10x + 3,000 - 6x$?

Step back to $\Pi = 10x + 12y$

Revenue from x is 40. Cost of x is 30. So the first term is clear. If we frame the decision as two independent choices, then this (30) is the marginal cost of x .

However, when we substitute $(.5(500 - x))$ for y we are framing y as a consequence of x not as an independent choice.

$$\Pi = 10x + 12y$$

$$\Pi = 10x + 12(.5(500 - x))$$

$$\Pi = 10x + 3,000 - 6x$$

Framed this way with y expressed as a function of x our choice of x incurs additional costs through its effect on y . $-6x$ is change in the revenue from y due to a unit change in x .

Think about the term ‘cost’.

- Cost is the resources we sacrifice to produce x
- On one hand, 30 is the value of resources we consume to make a unit of x . We had \(\$30 in our account that we gave up for the opportunity to make a unit of x .
- On the other hand we also give up the opportunity to make y , so we give up some future revenue.
- Is this a cost? Not in terms of consumption, and marginal cost.
- But it is something that we sacrifice. Put differently, not all sacrifices are captured by traditional accounting measures of ‘cost’, this is a reduction in *benefit* in the form of reduced revenue.

Builder of garages and sheds

A builder specializes in the construction of car garages and tool sheds. It has at most 80 hours available to frame the garages and tool sheds and at most 74 hours to roof the garages and tool sheds. It takes 10 hours to frame a garage and 7 hours to roof it. It takes 4 hours to frame a tool shed and 5 hours to roof it. If the builder makes a profit of \(\$570 per garage and \(\$300 per tool shed, how many can he build to receive a maximum profit?

Do the following:

1. Formulate the builder's decision as a linear program. You do not need to solve the program.
2. Sketch the feasible set. Be sure to label the axes carefully. On the graph, circle any point corresponding to values of the choice variables that you think are likely a to be optimal.

Choice variables:

```
from gekko import GEKKO  
m=GEKKO(remote=False)  
# define the variables  
x=m.Var(name="sheds",lb=0)  
x.value=1  
y=m.Var(name="garages",lb=0)  
y.value=1
```

Objective function:

```
m.Maximize(300*x+570*y)  
def profit(x,y):  
    return 300*x+570*y
```

Constraints:

```
m.Equation(4*x+10*y<80) # framing constraint  
m.Equation(5*x+7*y<74) # roofing constraint
```

<gekko.gekko.EquationObj at 0x1643f2bb0>

Solve!!!

Just for fun :)

```

m.solve(disp=False)
pi=profit(x.value[0],y.value[0])
print("x:      ", int(x.value[0]))
print("y:      ", int(y.value[0]))
print("Profit: ",int(pi))

```

```

x:      8
y:      4
Profit: 5149

```

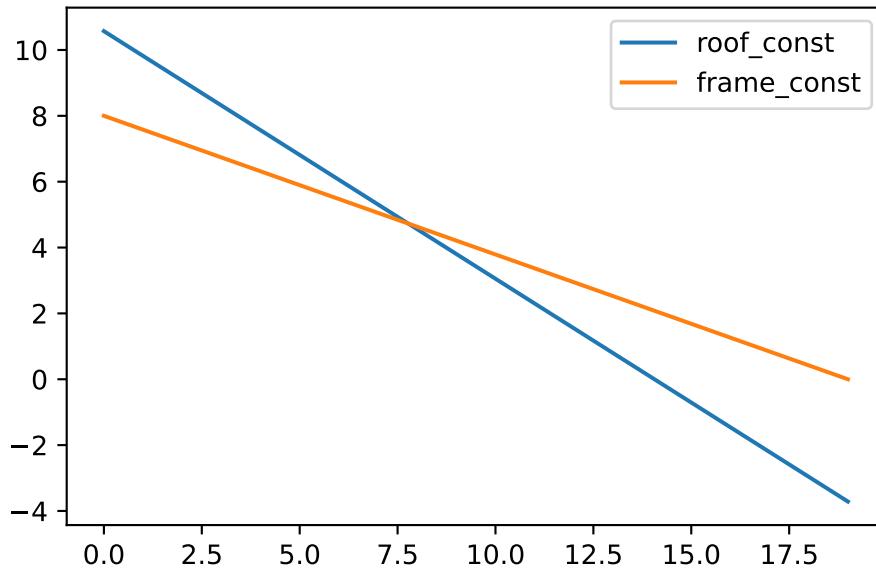
Plot

We can set this up as $y = f(x)$

```

# set our x axis
x=np.linspace(0,20,20)
# calculate the values of the constraints along x
frame_const=(80-4*x)/10
roof_const=(74-5*x)/7
# toss these arrays to pandas for a quick plot
pd.DataFrame({"roof_const":roof_const,"frame_const":frame_const}).plot()

```



Ava Catering

Setup:

Ava runs a full-service catering business where she caters banquets and receptions. She prepares food ahead of time and she sets up tables on the day of the event. Food preparation for banquets is 12 hours and for receptions is 4, while setup hours for banquets is 9 hours and receptions is 6 hours. The maximum hours available for food preparation are 35 hours and for setup are 40 hours. If she makes \\$800 for each banquet and \\$525 for each reception, how many banquets and receptions can Ava cater to make a maximum profit?

Python set up:

```
m=GEKKO(remote=False)
# 1 choices
x=m.Var(name="banquets",lb=0)
x.value=1
y=m.Var(name="receptions",lb=0)
y.value=1
# 2 objective
m.Maximize(800*x+525*y)
def profit(x,y):
    return 800*x+525*y
# 3 constraints
m.Equation(12*x+4*y<35) # prep time constraint
m.Equation(9*x+6*y<40) # set up constraint
```

<gekko.gekko.EquationObj at 0x166c45a90>

Python solution:

```
# 4 solve
m.solve(disp=False)
pi_1=profit(x.value[0],y.value[0])
print("x:      ", int(x.value[0]))
print("y:      ", int(y.value[0]))
print("Profit: ",int(pi_1))
```

x: 1
y: 4
Profit: 3517

Final questions:

1. What risks do you think a real business should consider that are not described here?
2. Describe how you would gather and analyze this information.
3. Based on the risks and the profit you expect, would you recommend opening a catering business to compete with Ava? Please describe your reasoning in detail.

Again, I am leaving these unanswered here so that you can spend some time thinking about how you would answer.