Homework #1

Jonathan Beaubien
Math 381 - Discrete Modeling
University of Washington

January 15th, 2021

Inquiry 1

Suppose we have 50 unique objects with corresponding weights, values, and volumes. Suppose object i (where $1 \leq i \leq 50$) has the following value, weight and volume:

$$\text{value}_i = \text{floor}(50 + 30cos(i)) \text{ (thousands of dollars)}$$

$$\text{weight}_i = \text{floor}(14 + 9cos(11i + 2)) \text{ (kg)}$$

$$\text{volume}_i = \text{floor}(10 + 2cos(4i - 1)) \text{ (liters)}$$

The floor function simply truncates any decimal places so that the value is an integer.
We have a 'knapsack' for these objects yet it can only contain a weight of 200 kg and a volume of 100 liters.
To maximize the total value, we will use a linear program to help us.
First, however, we must produce a linear program file (by way of the python programming language).
Let whether object i is in the 'knapsack' or not be represented by an integer: $x_i \in 0, 1$ where 0 denotes not included while 1 denotes included.

```
import math

f = open("problem_1.lp", "w")
f.write("max: ")
for i in range(1,51):
  f.write("+" + str(math.floor(50 + 30 * math.cos(i))) + "x" + str(i))
f.write(";\n")
for i in range(1,51):
  f.write("+" + str(math.floor(14 + 9 * math.cos(11 * i + 2))) + "x" + str(i))
f.write(" <= 200;\n")
for i in range(1,51):
  f.write("+" + str(math.floor(10 + 2 * math.cos(4 * i - 1))) + "x" + str(i))
f.write(" <= 100;\n")
f.write("bin x1")
for i in range(2,51):
  f.write(",x" + str(i))
f.write(";")
f.close()
```

This code produces a linear program file that allows use to maximize the value with the constrains of the 'knapsack'.

The file looks like this (ellipses for succinctness):

```
max: +66x1+37x2+20x3+30x4 ... +78x50;
+22x1+17x2+5x3+10x4+ ... +19x50 <= 200;
+8x1+11x2+10x3+8x4+ ... +9x50 <= 100;
bin x1,x2,x3,x4, ... ,x50;
```

After running our linear program into the program lpsolve, we get this output (all objects, x_i, with value 0 are not included):

```
Value of objective function: 840.00000000

Actual values of the variables:
x1                              1
x6                              1
x7                              1
x12                             1
x18                             1
x20                             1
x26                             1
x31                             1
x37                             1
x39                             1
x45                             1
x50                             1
```

It turns out our maximum value with the conditions of a max weight of 200 kg and max volume of 100 liters in \$840,000.

Inquiry 2

Suppose we have the same initial setup as in the first inquiry but we want the prime numbered objects $(1, 2, 3, 5, 7$,etc.) to make up 50% of the knapsack.
We will manipulate the code for the creation of the linear program in python to include something like this:

$$2 * \text{sum}(\text{value}(x_i)) \text{ where } i \text{ is prime} \geq \text{sum}(\text{value}(x_i))$$

```python
import math

def isPrime(n) :
  if (n==1):
    return False
  elif (n==2):
    return True
  else:
    for x in range(2,n):
      if(n % x==0):
        return False
    return True

f = open("problem_1.lp", "w")
f.write("max: ")
for i in range(1,51):
  f.write("+" + str(math.floor(50 + 30 * math.cos(i))) + "x" + str(i))
f.write(";\n")
for i in range(1,51):
  if isPrime(i):
    f.write("+" + str(2 * math.floor(50 + 30 * math.cos(i))) + "x" + str(i))
f.write(" >= ")
for i in range(1,51):
  f.write("+" + str(math.floor(50 + 30 * math.cos(i))) + "x" + str(i))
f.write(";\n")
for i in range(1,51):
  f.write("+" + str(math.floor(14 + 9 * math.cos(11 * i + 2))) + "x" + str(i))
f.write(" <= 200;\n")
for i in range(1,51):
  f.write("+" + str(math.floor(10 + 2 * math.cos(4 * i - 1))) + "x" + str(i))
f.write(" <= 100;\n")
f.write("bin x1")
for i in range(2,51):
  f.write(",x" + str(i))
f.write(";")
f.close()
```

We end up with the following abbreviated linear program:

```
max: +66x1+37x2+20x3+30x4+ ... +78x50;
+74x2+40x3+116x5+144x7+ ... +40x47 >= +66x1+37x2+20x3+30x4+ ... +78x50;
+22x1+17x2+5x3+10x4+ ... +19x50 <= 200;
+8x1+11x2+10x3+8x4+ ... +9x50 <= 100;
bin x1,x2,x3,x4, ... ,x50;
```

We end up with an output from lpsolve of this:

```
Value of objective function: 822.00000000

Actual values of the variables:
x6                              1
x7                              1
x12                             1
x13                             1
x19                             1
x25                             1
x26                             1
x31                             1
x37                             1
x43                             1
x50                             1
```

By requiring 50% of the value to come from prime numbered objects we only receive $822,000 within our constraints.
The math for calculating the same scenario but for 75% is relatively straightforward.

```
import math

def isPrime(n) :
  if (n==1):
    return False
  elif (n==2):
    return True
  else:
    for x in range(2,n):
      if(n % x==0):
        return False
    return True

f = open("problem_1.lp", "w")
f.write("max: ")
for i in range(1,51):
  f.write("+" + str(math.floor(50 + 30 * math.cos(i))) + "x" + str(i))
f.write(";\n")
for i in range(1,51):
  if isPrime(i):
    f.write("+" + str(4 * math.floor(50 + 30 * math.cos(i))) + "x" + str(i))
f.write(" >= ")
for i in range(1,51):
  f.write("+" + str(3 * math.floor(50 + 30 * math.cos(i))) + "x" + str(i))
f.write(";\n")
for i in range(1,51):
  f.write("+" + str(math.floor(14 + 9 * math.cos(11 * i + 2))) + "x" + str(i))
f.write(" <= 200;\n")
for i in range(1,51):
  f.write("+" + str(math.floor(10 + 2 * math.cos(4 * i - 1))) + "x" + str(i))
f.write(" <= 100;\n")
f.write("bin x1")
for i in range(2,51):
  f.write(",x" + str(i))
f.write(";")
f.close()
```

We end up with the following abbreviated linear program:

```
max: +66x1+37x2+20x3+30x4+ ... +78x50;
+148x2+80x3+232x5+288x7+ ... +80x47 >= +198x1+111x2+60x3+90x4+ ... +234x50;
+22x1+17x2+5x3+10x4+ ... +19x50 <= 200;
+8x1+11x2+10x3+8x4+ ... +9x50 <= 100;
bin x1,x2,x3,x4, ... ,x50;
```

We end up with an output from lpsolve of this:

```
Value of objective function: 729.00000000

Actual values of the variables:
x5                            1
x6                            1
x7                            1
x12                           1
x13                           1
x17                           1
x19                           1
x23                           1
x31                           1
x37                           1
x43                           1
```

By requiring 70% of the value to come from prime numbered objects we only receive $729,000 within our constraints.

Inquiry 3

Lastly, we will consider the case of maximizing the value of objects where the number of each object does not need to be $\{0, 1\}$ but any integer $\{0, 1, 2, 3, ...\}$.
We can do this by manipulating the python code to produce an augmented linear program. All we need to change is line 14 of the original program to include 'int' rather than 'bin':

```
1 f.write("int x1")
```

We get a new linear program file:

```
max: +66x1+37x2+20x3+30x4+ ... +78x50;
+22x1+17x2+5x3+10x4+ ... +19x50 <= 200;
+8x1+11x2+10x3+8x4+ ... +9x50 <= 100;
int x1,x2,x3,x4, ... ,x50;
```

Running this through lpsolve we get the output:

```
Value of objective function: 937.00000000

Actual values of the variables:
x6                           10
x19                           1
x50                           1
```

As we can see, object 6 must provide much value relative to its weight and volume. Our maximum value is now $937,000 which is much more value than the first scenario.