Gerald Jones, Kimon Swanson, Alex Hines

Home Work 5: Farmers Problem

ISE522 Spg 22

Notebook Links:

1. Data Display section
2. Problem 1
3. Problem 2
4. Problem 3

Problem Description:

> The farmer's problem Consider and farmer in east Tennessee who specializes in raising wheat, corn, and sugar beets on her 500 acres of land. During the winter, she wants to decide how much land to devote to each crop. The farmer knows that at least 200 tons (T) of wheat and 240 T of corn are needed for cattle feed. These amounts can be raised on the farm or bought from a wholesaler. Any production in excess of the feeding requirement would be sold. Over the last decade, mean selling prices have been $170 and $150 per ton of wheat and corn, respectively. The purchase prices are 40% more than this due to the wholesaler's margin and transportation costs.
>
> Another profitable crop is **sugar beet**, which he expects to sell at $36/T; however, the EU imposes a quota on sugar beet production. Any amount in excess of the quota can be sold onlv at $10/T. The farmer'squota for next year is 6,000 T. Based on past experience, the farmer knows that he mean yield on his land is roughly 2.5 T,3 T, and 20 T per acre for wheat, corn, and sugar beets, respectively. The table on the next slide summarizes these data and provides the planting costs for these crops. Formulate a linear program to maximize total profit (planting cost —revenue from sales).

Extentions To Accomplish:

1. Suppose the farmer allocates 120 acres to wheat, 80 to corn, and 300 for sugar beets. But also suppose that the yields are not certain but are subject to the same 3 scenarios we discussed in class (above average, average, below average, each with equation probability. What is the expected annual profit?
   - Problem 1 Method Defintion and Execution Cell
2. Consider an extension of the problem in which prices also are uncertain. Specifically, when yields are above average prices for corn and wheat go down by 10% and when yields are below average, they go up by 10%. Assume sugar beet prices are not affected by yields. Formulate and solve a stochastic linear program and solve it with a solver.
3. Consider a different version of the problem in which the farmer possesses four fields of sizes 185, 145, 105, and 65, respectively. The total of 500 acres is unchanged. However, the farmer wishes to only plant one type of crop on each field. Formulate and solve a stochastic program and solve it with a solver

Assumptions:

Problem 1:

Problem 2:

- the increase in prices for feed crops applies to after the first stage of planting only and thus the purchase prices, and selling prices are affected while the planting costs are not.

Problem 3:

Module imports and data loading

In [1]:
```python
from _GUROBI_TOOLS_.GUROBI_MODEL_BUILDING_TOOLS import *
from _NOTE_BOOK_UTILS import *
import numpy as np
import os
mdir = os.getcwd()
# name for the conversion to pdf method from _NOTE_BOOK_UTILS
notebook_title = mdir + "\\" + "_HW5_Problem3.ipynb"
```

Data Display & Python Variable Assignments

In [2]:
```python
# display data for problem
# data is represented by the instantiated dataframe show here and displayed in the output below

# data for the problem
data_df = pd.DataFrame(
{
    "Wheat":[2.5, 150, 170, 170, 238, 200],
    "Corn": [3,   230, 150, 150, 210, 240],
    "Sugar Beets":[20, 260, 36, 10, 0, 0],
},
    index=["Yield", "Planting Cost", "Selling Cost1", "Selling Cost2", "Price", "min"]
)

display(data_df)

# used to calculate the above/below average yields
average_rate = .2

# used to calculate the above/below average yields
average_rate = .2 # (percentage above/below the respective high and low average scenerios)

# price percentage difference in the low and high average yield scenerios
price_rate = .1



# names of the feed crops
feedCrops = ["Wheat","Corn"]

#
print("\t\t\t------------------------ Minimum crop resources for feed crops")
for c in feedCrops:
    print("Require a minimum of {} tons of {}".format(data_df.loc["min",c], c))
print("------------------------------------------------")

quota = 6000              # yield level that when exceeded causes price of sugar beets-->reduced/penalized
```

	Wheat	Corn	Sugar Beets
Yield	2.5	3	20
Planting Cost	150.0	230	260
Selling Cost1	170.0	150	36
Selling Cost2	170.0	150	10
Price	238.0	210	0
min	200.0	240	0

```
                        ------------------------ Minimum crop resources for feed crops
Require a minimum of 200.0 tons of Wheat
Require a minimum of 240 tons of Corn
------------------------------------------------
```

Method Definitions

Problem 1 Solver method

For this problem we just created a function to do the calculation. It can be found in the below cell.

Problem 1 Method Defintion and Execution Cell | Problem 1 Solution

In [3]:
```python
# calculate annual profit with yield uncertainty
def objective_uncertain_yields(df, xw, ysc, quota, minT_dict, scenerio_probs=None,
                               equally_likely=True, ):
    """
            df: dataframe containing columns Wheat, Corn, Sugar Beets, and indices Planting Cost, Yield,
            Selling Cost1, and Selling Cost2. The Yield,"Crop" is the expected yield rate of the
            given crop amount xw
            xw: list representing the amount of each crop that was planted. takes the form:
                    xw[0] --> amount of Wheat planted
                    xw[1] --> amount of Corn planted
                    xw[2] --> amount of Sugar Beets planted
            ysc: list of lists for the expected % of planted crop yielded. Takes the form:
                    ysc[scenerio, crop]
                    crop:  {0 (wheat), 1 (corn), 2 (Sugar Beets)}
                    scenerio: {0 (below average percentage of planted yielded),
                               1 (just 1 representing the expected average yield),
                               2 (above average percentage of planted yielded)}
    """
    #              ***** calculate and return annual costs under uncertainty *****


    if scenerio_probs is None:
        scenerio_probs = list()
        num_cases = len(ysc.shape[1])
        for i in range(num_cases):
            scenerio_probs.append(1.0/num_cases)

    # amount of each crop planted
    wheat_planted = xw[0]
    corn_planted = xw[1]
    sugarbeets_planted = xw[2]

    # average expected yields
    avg_wheat = df.loc["Yield", "Wheat"]
    avg_corn = df.loc["Yield", "Corn"]
    avg_sugarbeets = df.loc["Yield", "Sugar Beets"]

    # costs for planting the given acreages of each type of crop
    wheat_costs = df.loc["Planting Cost", "Wheat"]*wheat_planted
    corn_costs = df.loc["Planting Cost", "Corn"]*corn_planted
    sugarbeets_costs = df.loc["Planting Cost", "Sugar Beets"]*sugarbeets_planted

    # Total planting costs
    planting_costs = wheat_costs + corn_costs + sugarbeets_costs
    print("\t\t\t---------------Planting Costs for Given Crop Allotments---------------")
    print("\tWheat planting costs: {:>22.0f}".format(wheat_costs))
    print("\tCorn planting costs: {:>23.0f}".format(corn_costs))
    print("\tSugar beets planting costs: {:>16.0f}".format(sugarbeets_costs))
    print("\tTotal Planting Costs: {:>23.0f}".format(planting_costs))
    print("----------------------------------------------------------------------\n")

    # yield amounts for each scenerio
    # (below average= average + average*-below_average_rate = average(1 - rate))
    # (average = average + average*0 = average*(1 + 0)
    # (above average= average + average*above_average_rate = average(1 + rate))-->
    # make list for each crop where [below_average_yield, average_yield, above_average_yield]
    wheat_yields = [avg_wheat*(1+ysc[0, scenerio])*xw[0] for scenerio in range(len(ysc))]
    corn_yields = [avg_corn*(1+ysc[1, scenerio])*xw[1] for scenerio in range(len(ysc))]
    sugarbeets_yields = [avg_sugarbeets*(1+ysc[2, scenerio])*xw[2] for scenerio in range(len(ysc))]
    print("\t\t\t---------------Expected Yields (tons) for Each Possible Case---------------")
    print("\nWheat yields:\n\tBelow Average {:.2f},\n\tAverage {:.2f},\n\tAbove Average {:.2f}".format(
                                                                                wheat_yields[0],
                                                                                wheat_yields[1],
                                                                                wheat_yields[2]))

    print("\nCorn yields:\n\tBelow Average {:.2f},\n\tAverage {:.2f},\n\tAbove Average {:.2f}".format(
                                                                              corn_yields[0],
                                                                               corn_yields[1],
                                                                               corn_yields[2]))
    print("\nSugarbeet yields:\n\tBelow Average {:.2f},\n\tAverage {:.2f},\n\tAbove Average {:.2f}".format(
                                                                              sugarbeets_yields[0],
                                                                              sugarbeets_yields[1],
                                                                              sugarbeets_yields[2]))
    print("----------------------------------------------------------------------\n")

    # profits based on different scenerios
    # excess_wheat = a if a < b else b
    # use yields to:
    #         1) calculate the amount of wheat/corn that needs to be purchased
    #         2) calculate any excess wheat/corn that can be sold for profit

    #                 1) Amount to purchase for each scenerio
    excess_wheat, missing_wheat = list(), list()
    excess_corn, missing_corn = list(), list()
    excess_sugarbeets, justright_sugarbeets = list(), list()


    for wheatSi, cornSi, sugarbeetSi in zip(wheat_yields, corn_yields, sugarbeets_yields):
        # calculate the excess for scenerio i(below, average, above)
        excess_wheat.append(wheatSi - minT_dict["Wheat"] if wheatSi - minT_dict["Wheat"] >= 0 else 0 )
        missing_wheat.append(minT_dict["Wheat"] - wheatSi if minT_dict["Wheat"] - wheatSi >= 0 else 0)

        #
        excess_corn.append(cornSi - minT_dict["Corn"] if cornSi - minT_dict["Corn"] >= 0 else 0 )
        missing_corn.append(minT_dict["Corn"] - cornSi if minT_dict["Corn"] - cornSi >= 0 else 0)

        #
        excess_sugarbeets.append(sugarbeetSi - quota  if sugarbeetSi > quota else 0)
        justright_sugarbeets.append(quota  if sugarbeetSi > quota else sugarbeetSi)


    # Profits:
    wheat_profits = (np.array(excess_wheat) * df.loc["Selling Cost1", "Wheat"]).flatten()
    corn_profits = (np.array(excess_corn) *  df.loc["Selling Cost1", "Corn"]).flatten()
    sugarbeets_profits = np.array(excess_sugarbeets) * df.loc["Selling Cost2", "Sugar Beets"] \
                       + np.array(justright_sugarbeets) * df.loc["Selling Cost1", "Sugar Beets"]

    # Purchase Costs:
    print("\t\t\t---------------Expected Excess Crops for each case---------------")
    wm_fl = np.array(missing_wheat).flatten()
    we_fl = np.array(excess_wheat).flatten()
    cm_fl = np.array(missing_corn).flatten()
    ce_fl = np.array(excess_corn).flatten()

    print("Excess Wheat:\n\tBelow Average{:>23.0f},\n\tAverage{:>29.0f},\n\tAbove Average{:>23.0f}".format(
                                                                           we_fl[0],
                                                                           we_fl[1],
                                                                           we_fl[2]))
    print("Excess Corn:\n\tBelow Average{:>23.0f},\n\tAverage{:>29.0f},\n\tAbove Average{:>23.0f}".format(
                                                                           ce_fl[0],
                                                                           ce_fl[1],
                                                                           ce_fl[2]))

    print("----------------------------------------------------------------------\n")

    print("\t\t\t---------------Expected Crops That need to be purchased for each case---------------")
    print("\nWheat missing:\n\tBelow Average{:>23.0f},\n\tAverage{:>29.0f},\n\tAbove Average{:>23.0f}".format(
                                                                           wm_fl[0],
                                                                           wm_fl[1],
                                                                           wm_fl[2]))
    print("\nCorn missing:\n\tBelow Average{:>23.0f},\n\tAverage{:>29.0f},\n\tAbove Average{:>23.0f}".format(
                                                                           cm_fl[0],
                                                                           cm_fl[1],
                                                                           cm_fl[2]))



    # calculate the cost of purchasing needed wheat and corn
    wheat_purhased_costs = np.array(missing_wheat) * df.loc["Price", "Wheat"]
    corn_purhased_costs = np.array(missing_corn).flatten() * df.loc["Price", "Corn"]
    print("\t\t\t---------------Purchase Costs---------------")
    print("wheat purchased: ", wheat_purhased_costs)
    print("Corn purchased: ", corn_purhased_costs)
    print("----------------------------------------------------------------------\n")




    # Total purchasing costs
    purchasing_costs = wheat_purhased_costs + corn_purhased_costs

    print()
    print("\t\t\t---------------Sales profits---------------")
    print("wheat profits: ", wheat_profits)
    print("corn profits: ", np.array(corn_profits).flatten())
    # sugar beets to sell at the different levels of sales cost
    print("Sugar beets within quota: ", justright_sugarbeets)
    print("Sugar beets above quota: ", excess_sugarbeets)
    print("Sugar beets profits: ", sugarbeets_profits)
    # total profits from sales
    sale_profits = wheat_profits + corn_profits + sugarbeets_profits
    print("Sales Profits Total: ", sale_profits)
    print("----------------------------------------------------------------------\n")

    print("\t\t\t---------------Total Costs and Profits---------------")
    print("Sales Profits Total: ", sale_profits)
    print("Planting Costs: ", planting_costs)
    print("Purchasing Costs: ", purchasing_costs)
    print("----------------------------------------------------------------------\n")

    b_avg_obj = sale_profits[0] - planting_costs - purchasing_costs[0]
    avg_obj = sale_profits[1] - planting_costs - purchasing_costs[1]
    a_avg_obj = sale_profits[2] - planting_costs - purchasing_costs[2]
    print()
    print("\t\t\t---------------Expected Profits for Each Yield Scenerio---------------")
    print("Below Average Expected Profits: {}".format(b_avg_obj))
    print("Average Expected Profits: {}".format(avg_obj))
    print("Above Average Expected Profits: {}".format(a_avg_obj))
    print("----------------------------------------------------------------------\n")

    expected_profit = scenerio_probs[0]*b_avg_obj +\
                      scenerio_probs[1]*avg_obj +\
                      scenerio_probs[2]*a_avg_obj

    return (b_avg_obj, avg_obj, a_avg_obj),  expected_profit

##################################################################################
# Initial amount of acreage devoted to each crop type
crop_acreage = [
    120, # acres of wheat to plant
    80,  # acres of corn to plant
    300  # acres of sugar beets to plant
]

print("Initial Allotted Acreage:")
print("Wheat: {}".format(crop_acreage[0]))
print("Corn: {}".format(crop_acreage[1]))
print("Sugar Beets: {}".format(crop_acreage[2]))

# adjustment to yield rate based on scenerio
# rows-> crops, columns->scenerio
# In this problem the scenerios are expected to see a 20% decrease, or increase
# for the below and above average cases respectively
crop_yieldRate_scenerio = np.array([
                [-.2, 0, .2],  # below average, average, above average-->wheat
                [-.2, 0, .2],  # below average, average, above average-->corn
                [-.2, 0, .2],] # below average, average, above average-->sugar beets
              )

# Minimum amount of wheat/Corn that must be yielded for feed crop purposes
minT_dict = {
    "Wheat":200,        # minimum tons of wheat needed for feed
    "Corn":240,         # minimum tons of corn needed for feed
}

print("\t\t\t------------------------ Minimum crop resources for feed crops")
for c in minT_dict:
    print("Require a minimum of {} tons of {}".format(minT_dict[c], c))
print("------------------------------------------------")
# print(ysc)
# print(len(ysc))
quota = 6000              # yield level that when exceeded causes price of sugar beets-->reduced/penalized

# Probabilities for he below, average, and above average cases
scenerio_probs = [
    1/3,  # below average yield probability
    1/3,  # average yield probability
    1/3,  # above average yield probability
]

# call main method that performs calculation
three_cases, expected_profit =  objective_uncertain_yields(
                                                            data_df, #(Yield, Price, Selling-Prices)/Crop
                                                            crop_acreage,  # amount of acres devoted to each c
                                                            crop_yieldRate_scenerio, # yield rate adjustment s
                                                            quota, # sugar beets yield quota
                                                            minT_dict,  # key:=crop, val=minimum tons of crop
                                                            scenerio_probs, # probability of each scenerio
                                                            equally_likely=True,
                                                          )

# Parse below, average, and above average expected profits
below_average_case = three_cases[0]
average_case = three_cases[1]
above_average_case = three_cases[2]

# Print the calculated Expected cost based on the three cases for yield amounts
# and the given crop acreage allotments

print("Above average case Profit: {}".format(above_average_case))
print("Average case Profit: {}".format(average_case))
print("Below average case Profit: {}".format(below_average_case))
print("Expected Profit: {}".format(expected_profit))
```

```
Initial Allotted Acreage:
Wheat: 120
Corn: 80
Sugar Beets: 300
                        ------------------------ Minimum crop resources for feed crops
Require a minimum of 200 tons of Wheat
Require a minimum of 240 tons of Corn
------------------------------------------------
                        ---------------Planting Costs for Given Crop Allotments---------------
        Wheat planting costs:                  18000
        Corn planting costs:                   18400
        Sugar beets planting costs:            78000
        Total Planting Costs:                  114400
----------------------------------------------------------------------

                        ---------------Expected Yields (tons) for Each Possible Case---------------

Wheat yields:
        Below Average 240.00,
        Average 300.00,
        Above Average 360.00

Corn yields:
        Below Average 192.00,
        Average 240.00,
        Above Average 288.00

Sugarbeet yields:
        Below Average 4800.00,
        Average 6000.00,
        Above Average 7200.00
----------------------------------------------------------------------

                        ---------------Expected Excess Crops for each case---------------
Excess Wheat:
        Below Average                      40,
        Average                           100,
        Above Average                     160
Excess Corn:
        Below Average                       0,
        Average                             0,
        Above Average                      48
----------------------------------------------------------------------

                        ---------------Expected Crops That need to be purchased for each case---------------

Wheat missing:
        Below Average                       0,
        Average                             0,
        Above Average                       0

Corn missing:
        Below Average                      48,
        Average                             0,
        Above Average                       0
                        ---------------Purchase Costs---------------
wheat purchased:  [0. 0. 0.]
Corn purchased:  [10080.     0.     0.]
----------------------------------------------------------------------


                        ---------------Sales profits---------------
wheat profits:  [ 6800. 17000. 27200.]
corn profits:  [   0.    0. 7200.]
Sugar beets within quota:  [4800.0, 6000.0, 6000]
Sugar beets above quota:  [0, 0, 1200.0]
Sugar beets profits:  [172800. 216000. 228000.]
Sales Profits Total:  [179600. 233000. 262400.]
----------------------------------------------------------------------

                        ---------------Total Costs and Profits---------------
Sales Profits Total:  [179600. 233000. 262400.]
Planting Costs:  114400.0
Purchasing Costs:  [10080.     0.     0.]
----------------------------------------------------------------------


                        ---------------Expected Profits for Each Yield Scenerio---------------
Below Average Expected Profits: 55120.00000000001
Average Expected Profits: 118600.0
Above Average Expected Profits: 148000.0
----------------------------------------------------------------------

Above average case Profit: 148000.0
Average case Profit: 118600.0
Below average case Profit: 55120.00000000001
Expected Profit: 107240.0
```

Problem 1 Solution:

Problem 1 Method Defintion and Execution Cell The above code cell defines a function that calculates the expected profit when hedging on the three possible yield scenerios. The following sections will describe the three stages of the process and the resulting expected profit.

Solution: Expected Profits with Uncertainty Solution:

> To calculate the expected profit based on all three scenerios and the given acreage assignments each scenerios planting, purchasing costs are summed with the sales profits, then each scenerios profits are multiplied by the probability and these values are summed. The results of thes operations can be seen in the output labled "Expected Profit". From the calculation the hedged allotments would lead to an **expected profit of 107240.0.**

Stage 1: Deciding how much to plant

> For stage one the amounts to plant are given in the problem and are;
>
> - wheat = 120
> - corn = 80
> - sugar beets = 300

Attempting to Quantify Uncertainty into 3 scenerios:

> From the problem describe in class there are three possible scenerios of yields rates (acres/ton) that are considered.
>
> 1. The yield amount is below average i.e. 20% less than average
> 2. The yield amount is average i.e. what is found in the provided data table
> 3. The yield amount is above average i.e. 20% more than average Using the yield rates of 80% for below average, 100% for average, and 120% for above average each expected yield is increased by the corresponding yield case (below average, average, above average) to produce the expected yields seen in the above output labled, "Wheat yields", "Corn yields", "Sugar Beets yields" each with the corresponding yield case. These amounts can been seen in the above cells output under the "**Expected Yields (tons) for Each Possible Case**" section detailing the amount of tons of each crop is expected for each case.

Stage 2: How much needs to be purchased and sold

> The amount that needs to be purchased is the difference between the amounts of feed crops we have based on expected yield and the minimum amounts needed for feed crops. If the amount of feed crops (wheat/corn) yielded is below the minimum amount required for feed then the difference must be purchased (see "Expected Crops That need to be purchased for each case" section). If for a given scenerio the amount of feed crops are above the minimum amount the difference can be sold for a profit(see "Expected Excess Crops for each case" section). For each crop there is a set purchasing price and for each of the feed crops there is a set selling price. However, for the sugar beets there is a two level system of selling prices. If the amount of sugar beets yielded is above the set quota of 6kT then there is a higher sale price for each ton below this quota. Every ton above the quota must be sold at a lower selling price. If the amount sold is at or below the quota then only the higher price need be considered. If the amount of sugar beets yielded is in excess of the quota the difference between the quota and the amount of sugar beets yielded must be sold at the lower selling price. Thus the amount that needs to be sold in each scenerio is based on how much excess wheat, and corn we have, and the amount of sugar beets we have. The amount to be purhased is the difference between the minimum wheat and corn amounts and what is yielded in a given scenerio and only matters when there are less crops yielded than the minimum.

Calculating the Expected Yield:

Expected Profit

> To calculate the expected profit with the uncertainties you must calculate the expected profit in each of the scenerios and sum these expected profits weighted by their probability of occurance. For each scenerio the objective is:
>
> -planting_costs - purchasing_cost + sales_profits = gross_profit
> This Calculation must be performed for each yield scenerio. To accomplish this the method uses a set of matrices where the rows represent one of the crops and the columns represent one of the scenerios for the sales profits, and purchasing costs operations.

Planting Costs:

> The above defined and called method calcultes the planting costs for each crop and sums them to get the total planting costs displayed in the above output under the "Planting Costs for Given Crop Allotments" section of the output. These costs are constant across the scenerios because it is assumed that the given allotsments are used for each with differing yield conditions.

Sales Profits & Purchasing Costs:

Sales Profits:

> To calculate the sales profits two calculations are performed. First the amount of excess of each crop is calculated and the profit from the sale of this amount is calculated. For the sugar beets any amount of yield is considred excess and thus will be sold, but the amount it sells for is dependent on the yield for the scenerio. If the yeild is above the quota then the first 6K tons will be sold for the higher 36 ($/T) rate, and the remaining will be sold at the lower 10 ($/T) rates. For this step two arrays are kept. One represents the amount in each scenerio that is within the quota. If the yield for sugar beets does not exceed the quota then it is just this amount, if the yielded is above the quota then this value is set to the quota, and the excess is calculated and stored in the second array that holds the amount over the quota for a given scenerio. In a similar manner the amount in excess for the feed crops and the amount lacking and thus needing to be purchased based on the minimum are also kept as arrays. These arrays can be seen in the ouput labled missing_wheat, missing_corn, and justright_sugarbeets, for the arrays for the below, average, and above average cases for the amounts in each that are lacking in the feed crops and or that is below the quota for the sugar beets. The outputs labeled excess_wheat, excess_corn, and excess_sugarbeets are arrays of similar form for the amounts of extra wheat, and corn above the feed crop minimum is available for each scenerio and the amount of sugar beets over the quota repsectively. To calculate the sales profits in each scenerio the sales price for each crop type is multiplied by the excess crop arrays and the justright_sugarbeets arrays with the appropriate sales prices based on the table. This produces three arrays representing the below average, average, and above average cases for each of the three crops as seen in the "Sales profits" section of the output. At the end of the section an array labeled total sales profits is displayed that represents the total sales profits for each scenerio which is just the column wise summation of the three sales profits array for the crops.

Purchasing Costs:

> To calculate the purchasing costs the missing_wheat, and missing_corn arrays are multiplied by the purchasing costs for the corresponding crops. The results of this calculation can be seen in the output labeled "Purchase Costs".

Problem 2:

Model Formulation | Objective | Gurobi Implementation | Solution and Discussion

> Consider an extension of the problem in which prices also are uncertain. Specifically, when **yields are above average, prices for corn and wheat go down by 10%** and when **yields are below average, they go up by 10%**. **Assume sugar beet prices are not affected by yields**. Formulate and solve a stochastic linear program and solve it with a solver.

Model Formulation Problem 2

Parameters and Sets 2:

$\mathbf{C}$ set of crops $\{1, 2, 3, 4\}$ $c \in \mathbf{C}$

S expected level for yield $\{1, 2, 3\}$ where s $\in S$

μ_s probability of scenerio s occuring

X_c acres allocated to crop c, $\in \{1, 2, 3\}$

$Y_{s,c}$ tons purchased in scenerio s for crop c $\in \{1, 2\}$

$E_{s,c}$ Expected yield (tons) in scenerio s for crop c

$W_{s,c}$ tons of crop c sold in scenerio s

P_c purchase cost for crop $c \in \{wheat, corn\}$

L_c planting cost (\$/acre) for crop $c \in C$

$C_{s,c}$ purchase price for crop c in scenerion s, c $\in \{1, 2\}$

$F_{s,c}$ sale price for crop c

M_c minimum amount of feed crop c $\in 1, 2$

A maximum amount of acreage available

r_s the purchase price for yield scenerio s

$\rho_{s,c}$ average expected ton/acre yield rate for crop c for scenerio s

$\mathbb{G}$ Total Planting costs

$\mathbb{R}_s$ Total Purchasing costs for feed for scenerio s

$\mathbb{H}_s$ Total sales profits from all excess crops for scenerio s

$\mathbb{O}_s$ objective value for scenerio s

Equations and Constraints:

First Stage: How much to plant

Nonzero acreage constraint: $X_c \geq 0, \ \forall c$

Maximum Acreage Available: $\sum_c^{|C|}(X_c) \leq A$

Planting Costs: $\mathbb{G} = \sum_{c=1}^{3}(X_c \cdot L_c)$

Below Average:

Expected yield below average expected: $E_{1,c} = \rho_c(.8), \forall c$

purchase cost constraint based on yield:
$C_{1,c} == P_c \cdot (1.1) \implies E_{1,c} == \rho_c \cdot (.8),$ c $\in \{0, 1\}$

Total Sale Profits costs: $\mathbb{H}_1 = \sum_{c=1}^{2}(W_{1,c} \cdot F_{1,c} \cdot (1.1)) + \sum_{c=3}^{4}(W_{1,c} \cdot F_{1,c})$

Total Purchase costs: $\mathbb{R}_1 = \sum_{c=1}^{2}(Y_{1,c} \cdot C_{1,c})$

Average:

Expected yield below average expected: $E_{2,c} = \rho_c, \forall c$

purchase cost constraint based on yield:
$C_{2,c} == P_c \implies E_{2,c} == \rho_c,$ c $\in \{0, 1\}$

Total Sale Profits costs: $\mathbb{H}_2 = \sum_{c=1}^{4}(W_{2,c} \cdot F_{2,c})$

Total Purchase costs: $\mathbb{R}_2 = \sum_{c=1}^{2}(Y_{2,c} \cdot C_{2,c})$

Above Average:

Expected yield below average expected: $E_{3,c} = \rho_c(1.2), \forall c$

purchase cost constraint based on yield:
$C_{3,c} == P_c \cdot (.9) \implies E_{3,c} == \rho_c \cdot (1.2),$ c $\in \{0, 1\}$

Total Sale Profits costs: $\mathbb{H}_3 = \sum_{c=1}^{2}(W_{3,c} \cdot F_{3,c} \cdot (.9)) + \sum_{c=3}^{4}(W_{3,c} \cdot F_{3,c})$

Total Purchase costs: $\mathbb{R}_3 = \sum_{c=1}^{2}(Y_{3,c} \cdot C_{3,c})$

General Constraints:

Minimum feed crop constraints:
$Y_{s,c} + E_{s,c} - W_{s,c} \geq M_c, \forall s, c$ where c $\in$ {wheat(1), corn(2)}

Sugar Beets Sales constraint: $E_{s,3} - \sum_{c=3}^{4} W_{s,c} \geq 0, \forall s$

Sugar Beet below quota value constraint: $W_{s,2} \leq 6000, \forall s$

Objective:

> Maximize: $\sum_{s=1}^{3}((\mathbb{H}_s - \mathbb{R}_s) \cdot \mu_s) - \mathbb{G}$

Gurobi Implementation and Solution Problem 2

In [4]:
```python
try:
    # instantiate model object
    m = gp.Model("Farmer_problem2")



    ##########################################################################################
    ################################## Parameters set up #####################################
    ##########################################################################################
    C = 3       # number of crop lots
    S = 3       # number of possible scenerios
    F = len(feedCrops)        # amount of feed crops
    A = 500
    display(data_df)

    # Probabilities for he below, average, and above average cases
    scenerio_probs = [
        1/3,  # below average yield probability-->
        1/3,  # average yield probability
        1/3,  # above average yield probability
    ]
    crop_yieldRate_scenerio = [.8, 1, 1.2]  # below average, average, above average
    crop_price_scenerio =[1.1, 1, .9]  # below average, average, above average


    ##########################################################################################
    ################################## Variables set up ######################################
    ##########################################################################################
    Xc = m.addVars(C, vtype=GRB.CONTINUOUS, name="X", lb=0)    # acreage per crop for planting
    Esc = m.addVars(S, C, vtype=GRB.CONTINUOUS, name="E", lb=0,)  # expected yield for each crop & scenerio
    Ysc = m.addVars(S, F, vtype=GRB.CONTINUOUS, name="Y", lb=0)   # amount to purchase for feed crops
    Csc = m.addVars(S, F, vtype=GRB.CONTINUOUS, name="C", lb=0)   # purchase costs for feed crops & scenerio
    Wsc = m.addVars(S, C+1, vtype=GRB.CONTINUOUS, name="W", lb=0)   # amount to sell per crop

    G =   m.addVars(1, vtype=GRB.CONTINUOUS, name="G", lb=0,)     # planting costs
    Hs = m.addVars(S, vtype=GRB.CONTINUOUS, name="H", lb=0) # profits for sales in each scenerio for each crop
    Rs = m.addVars(S, vtype=GRB.CONTINUOUS, name="R", lb=0) # purchase costs for each scenerio for feed crops


    crops = data_df.columns.to_list()

    ##########################################################################################
    ################################## Objective set up ######################################
    ##########################################################################################

    # set up expected yields for each scenerio
    m.addConstrs(Esc[s, c] == data_df.loc["Yield", crops[c]]*crop_yieldRate_scenerio[s]*Xc[c]
                                                          for s in range(S) for c in range(C))

    # set up planting costs (G)
    exprs = 0
    for c in range(C):
        print(data_df.loc["Planting Cost", crops[c]])
        exprs += Xc[c]*data_df.loc["Planting Cost", crops[c]]
    m.addConstr(G[0]==exprs)

    # maximum acreage constraint
    tot_acres = 0
    for c in range(C):
        tot_acres += Xc[c]
    m.addConstr(tot_acres <= A)
    # ##################### Feed Crop Constraints
    # set up purchasing cost for different scenerios
    m.addConstrs(Csc[s, c] == data_df.loc["Price", crops[c]]*crop_price_scenerio[s]
                                                 for s in range(S) for c in range(F))

    # set up minimum feed crop constraints
    m.addConstrs(Ysc[s, c] - Wsc[s, c] + Esc[s, c] >= data_df.loc["min", crops[c]]
                                                 for s in range(S) for c in range(F))

    # set up purchasing costs expressions for each scenerio
    for s in range(S):
        exp = 0
        for c in range(F):
            exp += Ysc[s, c]*data_df.loc["Price", crops[c]]*crop_price_scenerio[s]
        m.addConstr(Rs[s]==exp)

    # set up sales profits expressions
    for s in range(S):
        exp = 0
        for c in range(2):
            exp += Wsc[s, c]*data_df.loc["Selling Cost1", crops[c]]*crop_price_scenerio[s]
        exp += Wsc[s, 2]*data_df.loc["Selling Cost1", crops[2]] \
             + Wsc[s, 3]*data_df.loc["Selling Cost2", crops[2]]
        m.addConstr(Hs[s]==exp)



    # ##################### Sugar Beet Constraints
    # set up sugar beet amounts for different levels
    m.addConstrs(Esc[s, 2] - Wsc[s, 2] - Wsc[s, 3] >= 0  for s in range(S))
    m.addConstrs(Esc[s, 2] <= quota  for s in range(S))             # set up lower level of quota sales constr


    # set up expressions for profits minus purchase costs for each scenario
    below_average_money_flow = (Hs[0] - Rs[0])*(1/3)
    average_money_flow = (Hs[1]  - Rs[1])*(1/3)
    above_average_money_flow = (Hs[2]  - Rs[2])*(1/3)

    ##########################################################################################
    ##################################     Objective     #####################################
    ##########################################################################################
    m.setObjective(below_average_money_flow + average_money_flow + above_average_money_flow \
                  - G[0], GRB.MAXIMIZE)

    ##########################################################################################
    ################################## SOLVE:OPTIMIZE ########################################
    ##########################################################################################
    ## Objective should be for(H(s) - R(s))

    m.optimize()

    ##########################################################################################
    ################################## Display Results #######################################
    ##########################################################################################
    displayDecisionVars(m, end_sentinel="6")

    print("\n-------------Does it make sense?----------------------")
    print('Obj: {:.2f}'.format(m.ObjVal))


# catch some math errors
except gp.GurobiError as e:
    print('Error code ' + str(e.errno) + ': ' + str(e))

except AttributeError:
    print('Encountered an attribute error')
```

```
Restricted license - for non-production use only - expires 2023-10-25
```

	Wheat	Corn	Sugar Beets
Yield	2.5	3	20
Planting Cost	150.0	230	260
Selling Cost1	170.0	150	36
Selling Cost2	170.0	150	10
Price	238.0	210	0
min	200.0	240	0

```
150.0
230
260
Gurobi Optimizer version 9.5.0 build v9.5.0rc5 (win64)
Thread count: 6 physical cores, 12 logical processors, using up to 12 threads
Optimize a model with 35 rows, 43 columns and 85 nonzeros
Model fingerprint: 0x2db5ebb0
Coefficient statistics:
  Matrix range     [1e+00, 3e+02]
  Objective range  [3e-01, 1e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [2e+02, 6e+03]
Presolve removed 28 rows and 28 columns
Presolve time: 0.00s
Presolved: 7 rows, 15 columns, 21 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    4.6213333e+33   6.000000e+30   4.621333e+03      0s
      10    1.0685067e+05   0.000000e+00   0.000000e+00      0s

Solved in 10 iterations and 0.01 seconds (0.00 work units)
Optimal objective  1.068506667e+05
X[0] 170.00000
X[1] 80.00000
X[2] 250.00000
E[0,0] 340.00000
E[0,1] 192.00000
E[0,2] 4000.00000
E[1,0] 425.00000
E[1,1] 240.00000
E[1,2] 5000.00000
E[2,0] 510.00000
E[2,1] 288.00000
E[2,2] 6000.00000
Y[0,0] 0.00000
Y[0,1] 48.00000
Y[1,0] 0.00000
Y[1,1] 0.00000
Y[2,0] 0.00000
Y[2,1] 0.00000
C[0,0] 261.80000
C[0,1] 231.00000
C[1,0] 238.00000
C[1,1] 210.00000
C[2,0] 214.20000
C[2,1] 189.00000
W[0,0] 140.00000
W[0,1] 0.00000
W[0,2] 4000.00000
W[0,3] 0.00000
W[1,0] 225.00000
W[1,1] 0.00000
W[1,2] 5000.00000
W[1,3] 0.00000
W[2,0] 310.00000
W[2,1] 48.00000
W[2,2] 6000.00000
W[2,3] 0.00000
G[0] 108900.00000
H[0] 170180.00000
H[1] 218250.00000
H[2] 269910.00000
R[0] 11088.00000
R[1] 0.00000
R[2] 0.00000
-------------------------------------



-------------Does it make sense?----------------------
Obj: 106850.67
```

Solution Discussion: Problem 2

> From the results with Gurobi, the optimal lot sizes(X) **for wheat, corn, and sugar beets are 170, 80, and 250 acres, respectively**. This lot-sizing leads to a profit of $**106850.67**.
>
> The **below-average scenario yields 340 tons of wheat, 192 tons of corn, and 4000 tons of sugar beets**. Because there are less than 6KT of sugar beets, it all will be sold at the higher sales price for this scenario. Due to the corn crop generating less than 240 tons of corn for feed, the farmer purchases (Y[0,1]) 48 tons of corn. Thus, this scenario leads to an expected purchasing cost (R[0]) of $11088.00. From sales of excess corn(140T) and beets(4KT) crops, there is a total profit of (H[0]) $170180.00.
>
> The **average scenario** for this lot-sizing leads to yields of 425, 240, 5000 tons of wheat, corn, and sugar beets, respectively. Therefore, there are 225 tons of wheat and 5K tons of sugar beets to sell for this scenario. There are no excess tons of corn, but there is just enough to meet the feed crop requirements, and so no crops need be purchased by the farmer in the average case. This case leads to no purchases necessary (Y[1,:]) thus no purchasing costs (R[1]) and only sales profits(H[1]) of $218250.00
>
> And finally, in the **above-average** scenario, there will be 510, 288, and 6K tons of wheat, corn, and sugar beets, respectively. These yields lead to an excess of 310 tons of wheat, 48 tons of corn, and 6K tons of sugar beets to sell. This lot allotment generates a sales profit (H[2]) of $269910.00. There is enough of each feed crop to meet the minimum requirements; therefore, there is no need to purchase any additional crops, leading to zero purchasing costs (R[2]).

Problem 3:

Model Formulation | Objective | Gurobi Implementation | Solution

> 1. Consider a different version of the problem in which the farmer possesses four fields of sizes 185, 145, 105, and 65, respectively. The total of 500 acres is unchanged. However, the farmer wishes to only plant one type of crop on each field. Formulate and solve a stochastic program and solve it with a solver

Model Formulation Problem 3

Indices, Parameters, and Sets 3:

$\mathbf{C}$ set of crops $\{1,2,3,4\}$ $c \in \mathbf{C}$

F set of possible planting fields, $\{1,2,3,4\} f \in F$

$D_{f,c}$ binary variable, 1 if field f will have crop c planted, 0 otherwise

$X_{f,c}$ acres allocated to crop c, in field f, $c \in \{1,2,3\}$

Y_c tons purchased for crop c $\in \{1,2\}$

E_c Expected yield (tons) for crop c in tons

W_c tons of crop c sold

C_c purchase cost for crop $c \in \{1,2\}$

L_c planting cost (\$/acre) for crop $c \in C$

P_c sale price for crop c

μ_c minimum amount of feed crop c $\in \{1,2\}$

A_f maximum amount of acreage available for field f

ρ_c average expected ton/acre yield rate for crop c

$\mathbb{G}$ Total Planting costs

$\mathbb{R}$ Total Purchasing costs for feed crops

$\mathbb{H}$ Total sales profits from all excess crops

M large number

> ## Equations and Constraints
>
> **Nonzero acreage constraint:** $X_{f,c} \geq 0, \forall f, c$
>
> **Nonzero acreage constraint:** $X_{f,c} \leq M \cdot D_{f,c}, \forall f, c$
>
> **Binary Planting Decision variables:** $\sum_{c=1}^{3}(D_{f,c}) \leq 1, \forall f$
>
> **Maximum Acreage Available:** $\sum_{f=1}^{4}\sum_{c}^{|C|}(X_{f,c}) \leq 500$
>
> **Expected Yield for a given crop:** $E_c = \sum_{f=1}^{4}(\rho_c \cdot X_{f,c}), c \in \{1,2,3\}$
>
> **Purchasing Costs for crop c:** $C_c = E_c \cdot P_c$
>
> **Total purchasing costs:** $\mathbb{R} = \sum_{c=1}^{2}(C_c), c \in \{1,2\}$
>
> **Total Sales Profits costs:** $\mathbb{H} = \sum_{c=1}^{4} W_c \cdot P_c$
>
> **Total Planting Costs Expression:** $\mathbb{G} = \sum_{f=1}^{4}\sum_{c=1}^{3}(X_{f,c} \cdot L_c)$
>
> **Minimum feed crop constraints:**
> $Y_c + E_c - W_c \geq \mu_c, \forall c \text{ where } c \in \{\text{wheat}(1), \text{corn}(2)\}$
>
> **Sugar Beets Sales constraint:** $E_3 - \sum_{c=3}^{4} W_c \geq 0, \forall s$
>
> **Sugar Beet below quota value constraint:** $W_3 \leq 6000, \forall s$

Objective 3:

> **Maximize:** $\mathbb{H} - \mathbb{R} - \mathbb{G}$

Gurobi Implementation and Solution Problem 3

In [5]:
```python
try:
    # instantiate model object
    m = gp.Model("Farmer_problem3")



    ##########################################################################################
    ################################## Parameters set up #####################################
    ##########################################################################################
    C = 3       # number of crop lots
    S = 3       # number of possible scenerios
    L = 4       # number of different lots for planting
    F = len(feedCrops)        # amount of feed crops
    A = 500
    # field:        0    1    2    3
    field_sizes = [185, 145, 105, 65]
    display(data_df)
    crop_yieldRate_scenerio = [.8, 1, 1.2]  # below average, average, above average
    crop_price_scenerio =[1.1, 1, .9]  # below average, average, above average
    ##########################################################################################
    ################################## Variables set up ######################################
    ##########################################################################################
    Xfc = m.addVars(L, C, vtype=GRB.CONTINUOUS, name="X", lb=0)  # acreage per crop for planting
    Dfc = m.addVars(L, C, vtype=GRB.BINARY, name="D", lb=0)   # Decision Variable field to plant what crop
    Ec = m.addVars(C, vtype=GRB.CONTINUOUS, name="E", lb=0,)  # expected yield for each crop & scenerio
    Yc = m.addVars(F, vtype=GRB.CONTINUOUS, name="Y", lb=0)   # amount to purchase for feed crops
    Wc = m.addVars(C+1, vtype=GRB.CONTINUOUS, name="W", lb=0) # amount to sell per crop
    G =    m.addVars(1, vtype=GRB.CONTINUOUS, name="G", lb=0,)     # planting costs
    H = m.addVars(1, vtype=GRB.CONTINUOUS, name="H", lb=0) # profits for sales in each scenerio for each crop
    R = m.addVars(1, vtype=GRB.CONTINUOUS, name="R", lb=0) # purchase costs for each scenerio for feed crops
    M = A*1.5                             # large number is just the total lot space times a half

    # get list of the crop names
    crops = data_df.columns.to_list()

    ##########################################################################################
    ################################## Objective set up ######################################
    ##########################################################################################
    # set up expected yields for each crop over all fields
    for c in range(C):
        tot_crop_c = 0
        for l in range(L):
            tot_crop_c += Xfc[l, c]
        # set this crops total Estimated yield to be the total acreage it was given time
        # the expected yield rate
        m.addConstr(Ec[c] == data_df.loc["Yield", crops[c]]*tot_crop_c)

    # set up planting costs (G)
    exprs = 0
    for l in range(L):
        for c in range(C):
            print(data_df.loc["Planting Cost", crops[c]])
            exprs += Xfc[l, c]*data_df.loc["Planting Cost", crops[c]]
    m.addConstr(G[0]==exprs)

    # add big M constraint for decision variable usage
    # for each of the possible crop selections
    for l in range(L):
        for c in range(C):
            m.addConstr(Xfc[l, c] <= M*Dfc[l,c])

    # add singular lot to crop assignment constraint
    for l in range(L):
        lot_binary_total = 0
        # for each lot sum all decision variables and make sure they do not go above 0
        for c in range(C):
            lot_binary_total += Dfc[l, c]
        m.addConstr(lot_binary_total <= 1)

    # maximum acreage constraint
    tot_acres = 0
    for l in range(L):
        for c in range(C):
            tot_acres += Xfc[l,c]
    m.addConstr(tot_acres <= A)

    # lot size constraints for each lot
    for l in range(L):
        lot_size = 0      # initial total lot size to zero
        # for each possible crop sum up the amount for this lot
        for c in range(C):
            lot_size += Xfc[l, c]
        # add the constraint that the total acres alloted
        # can not exceed what is available
        m.addConstr(lot_size <= field_sizes[l])



    # #####################                          Feed Crop Constraints
    # set up purchasing cost for different scenerios


    # set up minimum feed crop constraints
    m.addConstrs(Yc[c] - Wc[c] + Ec[c] >= data_df.loc["min", crops[c]]
                                                     for c in range(F))

    # set up purchasing costs expressions for each scenerio
    exp = 0
    for c in range(F):
        exp += Yc[c]*data_df.loc["Price", crops[c]]

    m.addConstr(R[0]==exp)

    exp = 0
    # set up the total profits expression for the feed crops
    for c in range(2):
        exp += Wc[c]*data_df.loc["Selling Cost1", crops[c]]

    # add expressions for the sugar beets at the different cost levels
    exp += Wc[2]*data_df.loc["Selling Cost1", crops[2]] \
         + Wc[3]*data_df.loc["Selling Cost2", crops[2]]

    m.addConstr(H[0]==exp)



    # #####################                          Sugar Beet Constraints
    m.addConstr(Ec[2] - Wc[2] - Wc[3] >= 0)   # set up sugar beet amounts constraint
    m.addConstr(Ec[2] <= quota)              # set up lower level of quota sales constraint

    # get each scenerios purchasing costs and sales profits


    ##########################################################################################
    ################################## Constraint set up #####################################
    ##########################################################################################
    m.setObjective(H[0] - R[0] - G[0], GRB.MAXIMIZE)

    ##########################################################################################
    ################################## SOLVE:OPTIMIZE ########################################
    ##########################################################################################
    ## Objective should be for(H(s) - R(s))

    m.optimize()

    ##########################################################################################
    ################################## Display Results #######################################
    ##########################################################################################
    displayDecisionVars(m, end_sentinel="10")

    print("\n-------------Does it make sense?----------------------")
    print('Obj: {:.2f}'.format(m.ObjVal))


# catch some math errors
except gp.GurobiError as e:
    print('Error code ' + str(e.errno) + ': ' + str(e))

except AttributeError:
    print('Encountered an attribute error')
```

	Wheat	Corn	Sugar Beets
Yield	2.5	3	20
Planting Cost	150.0	230	260
Selling Cost1	170.0	150	36
Selling Cost2	170.0	150	10
Price	238.0	210	0
min	200.0	240	0

```
150.0
230
260
150.0
230
260
150.0
230
260
150.0
230
260
Gurobi Optimizer version 9.5.0 build v9.5.0rc5 (win64)
Thread count: 6 physical cores, 12 logical processors, using up to 12 threads
Optimize a model with 31 rows, 36 columns and 106 nonzeros
Model fingerprint: 0x922aa527
Variable types: 24 continuous, 12 integer (12 binary)
Coefficient statistics:
  Matrix range     [1e+00, 8e+02]
  Objective range  [1e+00, 1e+00]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 6e+03]
Found heuristic solution: objective -30400.00000
Presolve removed 12 rows and 14 columns
Presolve time: 0.00s
Presolved: 19 rows, 22 columns, 54 nonzeros
Variable types: 14 continuous, 8 integer (8 binary)

Root relaxation: objective 1.186000e+05, 8 iterations, 0.00 seconds (0.00 work units)

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0 118600.000    0    2 -30400.000 118600.000   490%     -    0s
H    0     0                    65200.000000 118600.000  81.9%     -    0s
H    0     0                    72000.000000 118600.000  64.7%     -    0s
H    0     0                    99225.000000 118600.000  19.5%     -    0s
H    0     0                    106425.00000 118600.000  11.4%     -    0s
     0     0 118600.000    0    3 106425.000 118600.000  11.4%     -    0s
H    0     0                    107975.00000 118600.000  9.84%     -    0s
H    0     0                    114875.00000 118600.000  3.24%     -    0s
     0     0 114875.000    0    2 114875.000 114875.000  0.00%     -    0s

Cutting planes:
  MIR: 1
  Flow cover: 3

Explored 1 nodes (24 simplex iterations) in 0.02 seconds (0.00 work units)
Thread count was 12 (of 12 available processors)

Solution count 5: 114875 107975 106425 ... -30400

Optimal solution found (tolerance 1.00e-04)
Best objective 1.148750000000e+05, best bound 1.148750000000e+05, gap 0.0000%
X[0,0] 0.00000
X[0,1] 0.00000
X[0,2] 185.00000
X[1,0] 145.00000
X[1,1] 0.00000
X[1,2] 0.00000
X[2,0] 0.00000
X[2,1] 0.00000
X[2,2] 105.00000
X[3,0] 0.00000
X[3,1] 65.00000
X[3,2] 0.00000
D[0,0] 0.00000
D[0,1] -0.00000
D[0,2] 1.00000
D[1,0] 1.00000
D[1,1] 0.00000
D[1,2] 0.00000
D[2,0] 0.00000
D[2,1] 0.00000
D[2,2] 1.00000
D[3,0] 0.00000
D[3,1] 1.00000
D[3,2] 0.00000
E[0] 362.50000
E[1] 195.00000
E[2] 5800.00000
Y[0] 0.00000
Y[1] 45.00000
W[0] 162.50000
W[1] 0.00000
W[2] 5800.00000
W[3] 0.00000
G[0] 112100.00000
H[0] 236425.00000
R[0] 9450.00000
-------------------------------------



-------------Does it make sense?----------------------
Obj: 114875.00
```

Solution 3 Exploration & Discussion

Solution

> From the results with Gurobi, the optimal lot sizes(X) and field allocations are **plant 145 acres of wheat in lot 2, plant 65 acres of corn in lot 4, plant 185 acres of sugar beets in lot 1 and 105 in lot 3**. This lot-sizing and field allocation scheme leads to an optimal profit value of $**114875**.
>
> The cell above shows the solution in a condensed form and points out several things about the solution:
>
> 1. The given solution will require the purchase of 65 tons of corn due to the yield being below the minimum requirement. This leads to an extra purchasing cost of 9450.
> 2. Due to there being below the quota of sugar beets yielded they can all be sold at the higher price point.

acreage Decisions:

- Lot 1:
  - X[0,2] 185.00000 --> Sugar Beets
- Lot 2:
  - X[1,0] 145.00000 --> Wheat
- Lot 3:
  - X[2,2] 105.00000 --> Sugar beets
- Lot 4:
  - X[3,1] 65.00000 --> Corn
- Total acreage used: 185 + 145 + 105 + 65 = 500 = MAX Available

Planting Costs:

- G[0] = 112100.00000

Expected Yields:

- E[0] = 362.50000 --> no need to buy any wheat, excess of 362.5 - 200 = 162.5
- E[1] = 195.00000 --> **need to buy 240-195=45 tons of corn**
- E[2] = 5800.00000 --> **have below the quota of sugar beets so do not need to sale any at reduced price**

amounts of feed crops to purchase:

Y[0] = 0.00000 --> wheat Y[1] = 45.00000 --> corn (Matches above)

Purchasing costs:

- R[0] = 9450.00000 --> should be 45*210=9,450, thus the answer checks out

Tons of Excess crops:

W[0] = 162.50 --> wheat W[1] 0.00000 --> Corn W[2] 5800.00000 --> Beets at highest price W[3] 0.00000 --> Beets at lowest price

Profits from crop sales:

- H[0] = 236425.00000
- have 162 excess wheat so should make = 162*170=27,625
- have 5800 beets to sell so should make = 5800*36=208800
- Total profits: 27625 + 208880 = 236,425

Objective Check

- Thus H - G - R = 236425 - 112100 - 9450 = 114875, which is what the given objective was thus at least the math works out.

In [ ]:
```python
# save the notebook as a pdf
# to_PDF(notebook_title)
```