MIE 1620 Computational Project

Method-3:

The third method is to solve by Gurobi solver.

The original Problem:	The Dual/Block Angular structure
For all scenarios=S Decision variables: x0[s] = acres of land devoted to wheat x1[s] = acres of land devoted to corn x2[s]= acres of land devoted to sugar beets yw [s]= tons of wheat purchased yc[s] =tons of corn purchased zw[s]=tons of wheat sold zc[s]= tons of corn sold zb[s]= tons of sugar beets sold at favorable price zbb[s]= tons of sugar beets sold at lower price	For all scenarios=S Decision variables(prices for the original) x y1[s] y2[s] y3[s] y4[s]

Results:

Scenario=3	

Original [L-shaped] LP
 	
Model file:
 	
3.Farmer_model(S=3).lp
 	
Code file:
 	
3.Farmer-ted with Gurobi(S=3).py
 	
Optimal objective -1.083900000e+05
 	
Optimal solution
`x[0] 170`
`x[1] 80`
`x[2] 250`
`yw[0] 0`
`yw[1] 0`
	Dual of the problem
	[Block Angular structure]
	
	Model file:
	
	3.Dual(S=3).lp
	
	Code file:
	
	3.Dual (S=3).py
	
	Optimal objective 1.083900000e+05
	
	Optimal solution
	`x 275`
	`y1[1] 56.6667`
	`y1[2] 56.6667`
	`y1[3] 56.6667`

<table>
<tr><td>yw[2] 0
yc[0] 0
yc[1] 0
yc[2] 48
zc[0] 48
zc[1] 0
zc[2] 0
zb[0] 6000
zb[1] 5000
zb[2] 4000
zw[0] 310
zw[1] 225
zw[2] 140
zbb[0] 0
zbb[1] 0
zbb[2] 0</td><td>y2[1] 50
y2[2] 52.3333
y2[3] 70
y3[1] 4.29167
y3[2] 12
y3[3] 12
y4[1] 7.70833
y4[2] 0
y4[3] 0</td></tr>
<tr><td>Scenario=5</td><td></td></tr>
<tr><td><u>Original [L-shaped] LP</u>
<u>Model file:</u>
3.Farmer_model(S=5).lp

<u>Code file:</u>

3.Farmer-ted-gurobi(S=5).py

Optimal objective -1.024220000e+05

<u>Optimal solution</u>

x[0] 170
x[1] 80
x[2] 250
yw[0] 0
yw[1] 0
yw[2] 0
yw[3] 0
yw[4] 0
yc[0] 0
yc[1] 0
yc[2] 0
yc[3] 48
yc[4] 96
zw[0] 395
zw[1] 310
zw[2] 225</td><td><u>Dual of the problem</u>
<u>[Block Angular structure]</u>
<u>Model file:</u>
3.Dual(S=5).lp

<u>Code file:</u>
3.Dual (S=3).py

Optimal objective 1.485600000e+06

<u>Optimal solution</u>
x 3140
y1[1] 34
y1[2] 34
y1[3] 34
y1[4] 34
y1[5] 34
y2[1] 42
y2[2] 42
y2[3] 42
y2[4] 42
y2[5] 42
y3[1] 7.2
y3[2] 7.2</td></tr>
</table>

zw[3] 140 zw[4] 55 zc[0] 96 zc[1] 48 zc[2] 0 zc[3] 0 zc[4] 0 zb[0] 6000 zb[1] 6000 zb[2] 5000 zb[3] 4000 zb[4] 3000 zbb[0] 1000 zbb[1] 0 zbb[2] 0 zbb[3] 0 zbb[4] 0	y3[3] 7.2 y3[4] 7.2 y3[5] 7.2 y4[1] 0 y4[2] 0 y4[3] 0 y4[4] 0 y4[5] 0
Scenarios=9	
<u>Original [L-shaped] LP</u> <u>Model file:</u> Farmer_model3(S=9).lp <u>Code file:</u> 3.Farmer-ted-gurobi(S=9).py Optimal objective -9.178805556e+04 x[0] 232.5 x[1] 80 x[2] 187.5 yw[0] 0 yw[1] 0 yw[2] 0 yw[3] 0 yw[4] 0 yw[5] 0 yw[6] 0 yw[7] 0 yw[8] 83.75 yc[0] 0 yc[1] 0 yc[2] 0 yc[3] 0 yc[4] 0 yc[5] 48 yc[6] 96	 <u>Model file:</u> 3.Dual(S=9).lp <u>Code file:</u> 3.Dual (S=9).py Optimal objective 9.067793210e+04 x 278.778 y1[1] 18.8889 y1[2] 18.8889 y1[3] 18.8889 y1[4] 18.8889 y1[5] 18.8889 y1[6] 18.8889 y1[7] 18.8889 y1[8] 18.8889 y1[9] 26.4444 y2[1] 0.884774 y2[2] 23.3333 y2[3] 23.3333 y2[4] 23.3333 y2[5] 23.3333 y2[6] 23.3333 y2[7] 23.3333 y2[8] 23.3333

yc[7] 144 yc[8] 192 zw[0] 846.25 zw[1] 730 zw[2] 613.75 zw[3] 497.5 zw[4] 381.25 zw[7] 32.5 zw[8] 0 zc[0] 192 zc[1] 144 zc[2] 96 zc[3] 48 zc[4] 0 zc[5] 0 zc[6] 0 zc[7] 0 zc[8] 0 zb[0] 6000 zb[1] 6000 zb[2] 5250 zb[3] 4500 zb[4] 3750 zb[5] 3000 zb[6] 2250 zb[7] 1500 zb[8] 750 zbb[0] 750 zbb[1] 0 zbb[2] 0 zbb[3] 0 zw[5] 265 zw[6] 148.75 zbb[4] 0 zbb[5] 0 zbb[6] 0 zbb[7] 0 zbb[8] 0	y2[9] 23.3333 y3[1] 1.11111 y3[2] 1.58681 y3[3] 4 y3[4] 4 y3[5] 4 y3[6] 4 y3[7] 4 y3[8] 4 y3[9] 4 y4[1] 2.88889 y4[2] 2.41319 y4[3] 0 y4[4] 0 y4[5] 0 y4[6] 0 y4[7] 0 y4[8] 0 y4[9] 0

Method-1:

The first method to use is the simplex method as developed in class.

The original Problem:	The Dual/Block Angular structure
For all scenarios=S Decision variables: x0[s] = acres of land devoted to wheat x1[s] = acres of land devoted to corn x2[s]= acres of land devoted to sugar beets yw [s]= tons of wheat purchased yc[s] =tons of corn purchased zw[s]=tons of wheat sold zc[s]= tons of corn sold zb[s]= tons of sugar beets sold at favorable price zbb[s]= tons of sugar beets sold at lower price slack variables: ux uw[s] uc[s] ub[s] ubb[s] Artificial variables: vx[s] vw[s] vc[s] vb[s] vbb[s]	For all scenarios=S Decision variables(prices for the original) x y1[s] y2[s] y3[s] y4[s] slack variables: l[0],l[1],l[2] s1[s] s2[s] s3[s] s4[s] s5[s] s6[s] Artificial variables: la[0],la[1],la[2] a1[s] a2[s] a3[s] a4[s] a5[s] a6[s]

Results:

Scenario=3	

Original [L-shaped] LP
 	
Model file:
1.Farmer_model(S=3).lp
 	
Code file:
1.Farmersted -simplex -model1(S=3).py
 	
`Z=Z1+Z2`
`Z1= -108390.0`
`Z2 119969373520.36111`
	Dual of the problem
	[Block Angular structure]
	Model file:
	1.DualLP-simplex(S=3).lp
	
	Code file:
	
	1.Dual LP-simplex(S=3).py
	
	`Z1= 108390.00000000003`

<u>Optimal solution</u> `XB= [140.  170.  310.  225.   80.  48.   48. 6000. 5000. 4000.  250. 1000.  2000.]`	<u>Optimal solution</u> `XB= [56.66666667  52.33333333   0.95833333   2.33333333  20.  22.66666667 275.          70.  22.66666667  20.  17.66666667  22.66666667  50.  8.66666667  56.66666667  56.66666667   8.66666667   7.70833333   4.29166667  12.  12.        ]`
Scenario=5	

<u>Original [L-shaped] LP</u>
<u>Model file:</u>
3.Farmer_model(S=5).lp
 	
<u>Code file:</u>
 	
1.Farmersted -simplex-model1(S=5)
 	
`Z= -102421.99999999997`
 	
<u>Optimal solution</u>
 	
`XB= [48.  170. 2000.  310.  140. 1000.   80.   96.  395.  225.   96. 6000.  6000. 5000. 4000. 3000.  250. 1000.   55.   48. 3000.]`
	<u>Dual of the problem</u>
	<u>[Block Angular structure]</u>
	<u>Model file:</u>
	1.DualLP-simplex(S=5).lp
	
	<u>Code file:</u>
	
	1.DualLP-simplex(S=5).py
	
	`Z1= 244216.6666666666`
	
	<u>Optimal solution</u>
	`XB= [56.66666667 50. 20. 5.79166667 22.66666667 22.66666667 22.66666667 22.66666667 558.33333333 50. 70. 50.97222222 0.97222222 56.66666667 22.66666667 56.66666667 19.02777778 20. 50. 56.66666667 8.66666667 20. 20. 56.66666667 2.875 8.66666667 8.66666667 8.66666667 3.33333333 6.20833333 12. 12. 12.]`

Scenarios=9	

Original [L-shaped] LP

Model file:
1.Farmer_model(S=9).lp

Code file:
1.Farmer_model(S=9).lp

```
Z= -91788.05555555553
```

Optimal solution:

```
x[0] 232.5
x[1] 80
x[2] 187.5
yw[0] 0
yw[1] 0
yw[2] 0
yw[3] 0
yw[4] 0
yw[5] 0
yw[6] 0
yw[7] 0
yw[8] 83.75
yc[0] 0
yc[1] 0
yc[2] 0
yc[3] 0
yc[4] 0
yc[5] 48
yc[6] 96
yc[7] 144
yc[8] 192
zw[0] 846.25
zw[1] 730
zw[2] 613.75
zw[3] 497.5
zw[4] 381.25
zw[7] 32.5
zw[8] 0
zc[0] 192
zc[1] 144
zc[2] 96
zc[3] 48
zc[4] 0
zc[5] 0
zc[6] 0
zc[7] 0
zc[8] 0
zb[0] 6000
zb[1] 6000
zb[2] 5250
zb[3] 4500
```

Model file:
1.DualLP-simplex(S=9).lp

Code file:
1.DualLP-simplex(S=9)

objective `Z1= 486933.33333333343`

```
Z2 5.6999958860108005e+19
Z1=XB*CB
Z2=XN*CN
```

Solution

```
XB= [  56.66666667   50.
12.            50.            50.
   22.66666667   22.66666667   22.
66666667   22.66666667   22.666666
67
    1.66666667   22.66666667   50.
50.            20.
   50.           22.66666667   20.
1159.           79.33333333
   79.33333333   56.66666667   56.
66666667   20.            5.416666
67
   22.66666667   20.           18.
33333333   20.            20.
    8.66666667    8.66666667   56.
66666667   20.           56.666666
67
   20.           51.66666667   70.
70.            56.66666667
   22.66666667    3.25          8.
66666667    8.66666667    8.666666
67
    8.66666667    8.66666667    8.
66666667    3.33333333    3.333333
33
    6.58333333   12.           56.
66666667   12.            12.
   12.           12.         ]
```

zb[4] 3750 zb[5] 3000 zb[6] 2250 zb[7] 1500 zb[8] 750 zbb[0] 750 zbb[1] 0 zbb[2] 0 zbb[3] 0 zw[5] 265 zw[6] 148.75 zbb[4] 0 zbb[5] 0 zbb[6] 0 zbb[7] 0 zbb[8] 0	

The results of Simplex method showed that the model for 9 scenarios was not giving an optimal solution.
So for these large scale problems we need to implement another algorithm which is used when the matrix A has block angular structure i.e. the dual problem in this report.

<u>Method-2</u>

The second method to use is the Danzig Wolfe Decomposition as developed in class.

<u>Model file</u>:
Dantzewolfe.lp

<u>Code file:</u>
Dantzig wolfedecomposition.py

There are many errors in the code so could not solve large problems
So tried solving scenario=3 by hand and also did subproblems in gurobi solver at iterations.

<u>Subproblems file:</u>

subproblems(S=3).py